

libstdc++

Generated by Doxygen 1.9.7

1 Deprecated List	1
2 Todo List	2
3 Module Documentation	4
3.1 Algorithms	4
3.1.1 Detailed Description	4
3.1.2 Generalized Numeric operations	4
3.1.3 Mutating	20
3.1.4 Non-Mutating	43
3.1.5 Sorting	62
3.2 Atomics	105
3.2.1 Detailed Description	110
3.2.2 Macro Definition Documentation	110
3.2.3 Typedef Documentation	110
3.2.4 Enumeration Type Documentation	117
3.2.5 Function Documentation	117
3.3 Concurrency	117
3.3.1 Detailed Description	118
3.3.2 Condition Variables	118
3.3.3 Futures	119
3.3.4 Mutexes	122
3.3.5 Threads	125
3.4 Containers	126
3.4.1 Detailed Description	127
3.4.2 Associative	127
3.4.3 Sequences	129
3.4.4 Unordered Associative	130
3.5 Diagnostics	130
3.5.1 Detailed Description	131
3.5.2 Function Documentation	131
3.5.3 Exceptions	134
3.6 Extensions	140
3.6.1 Detailed Description	140
3.6.2 Dynamic Bitset.	140
3.6.3 Policy-Based Data Structures	145
3.6.4 SGI	155
3.7 File System	163
3.7.1 Detailed Description	168
3.7.2 Typedef Documentation	168

3.7.3 Enumeration Type Documentation	168
3.7.4 Function Documentation	168
3.8 I/O	169
3.8.1 Detailed Description	170
3.8.2 Typedef Documentation	171
3.9 Iterators	173
3.9.1 Detailed Description	177
3.9.2 Function Documentation	177
3.9.3 Iterator Tags	179
3.10 Locales	179
3.10.1 Detailed Description	181
3.10.2 Function Documentation	181
3.11 Numerics	182
3.11.1 Detailed Description	183
3.11.2 Bit manipulation	183
3.11.3 Complex Numbers	186
3.11.4 Decimal Floating-Point Arithmetic	196
3.11.5 Mathematical Special Functions	197
3.11.6 Mathematical constants	225
3.11.7 Numeric Arrays	225
3.11.8 Random Number Generation	242
3.11.9 TR1 Mathematical Special Functions	261
3.12 Ranges	267
3.13 Regular Expressions	267
3.13.1 Detailed Description	272
3.13.2 Typedef Documentation	272
3.13.3 Function Documentation	273
3.13.4 Base and Implementation Classes	301
3.14 Strings	302
3.14.1 Detailed Description	302
3.14.2 Typedef Documentation	302
3.15 Technical Specifications	303
3.15.1 Detailed Description	303
3.15.2 Filesystem TS	303
3.15.3 Library Fundamentals TS	309
3.15.4 Parallelism TS	322
3.16 Utilities	333
3.16.1 Detailed Description	340
3.16.2 Function Documentation	340

3.16.3 Variable Documentation	350
3.16.4 Concepts	350
3.16.5 Function Objects	351
3.16.6 Memory	361
3.16.7 Metaprogramming	388
3.16.8 Rational Arithmetic	406
3.16.9 Time	408
4 Namespace Documentation	424
4.1 <code>__gnu_cxx</code> Namespace Reference	424
4.1.1 Detailed Description	439
4.1.2 Typedef Documentation	439
4.1.3 Function Documentation	439
4.2 <code>__gnu_cxx::__detail</code> Namespace Reference	449
4.2.1 Detailed Description	450
4.2.2 Function Documentation	450
4.3 <code>__gnu_cxx::typelist</code> Namespace Reference	450
4.3.1 Detailed Description	451
4.3.2 Function Documentation	451
4.4 <code>__gnu_debug</code> Namespace Reference	451
4.4.1 Detailed Description	457
4.4.2 Typedef Documentation	457
4.4.3 Enumeration Type Documentation	457
4.4.4 Function Documentation	457
4.5 <code>__gnu_internal</code> Namespace Reference	459
4.5.1 Detailed Description	459
4.6 <code>__gnu_parallel</code> Namespace Reference	460
4.6.1 Detailed Description	467
4.6.2 Typedef Documentation	467
4.6.3 Enumeration Type Documentation	468
4.6.4 Function Documentation	469
4.6.5 Variable Documentation	508
4.7 <code>__gnu_pbds</code> Namespace Reference	508
4.7.1 Detailed Description	510
4.8 <code>__gnu_sequential</code> Namespace Reference	510
4.8.1 Detailed Description	510
4.9 <code>abi</code> Namespace Reference	510
4.9.1 Detailed Description	510
4.10 <code>std</code> Namespace Reference	510

4.10.1 Detailed Description	632
4.10.2 Typedef Documentation	632
4.10.3 Enumeration Type Documentation	634
4.10.4 Function Documentation	635
4.10.5 Variable Documentation	710
4.11 std::__debug Namespace Reference	711
4.11.1 Detailed Description	719
4.11.2 Function Documentation	719
4.12 std::__detail Namespace Reference	720
4.12.1 Detailed Description	723
4.12.2 Function Documentation	723
4.13 std::__parallel Namespace Reference	724
4.13.1 Detailed Description	741
4.13.2 Function Documentation	741
4.14 std::chrono Namespace Reference	741
4.14.1 Detailed Description	745
4.15 std::decimal Namespace Reference	745
4.15.1 Detailed Description	753
4.15.2 Function Documentation	754
4.16 std::experimental Namespace Reference	754
4.16.1 Detailed Description	764
4.16.2 Function Documentation	764
4.16.3 Variable Documentation	766
4.17 std::filesystem Namespace Reference	766
4.17.1 Detailed Description	769
4.18 std::literals Namespace Reference	769
4.18.1 Detailed Description	770
4.19 std::literals::chrono_literals Namespace Reference	770
4.19.1 Detailed Description	771
4.20 std::numbers Namespace Reference	771
4.20.1 Detailed Description	772
4.20.2 Variable Documentation	772
4.21 std::placeholders Namespace Reference	773
4.21.1 Detailed Description	774
4.22 std::regex_constants Namespace Reference	774
4.22.1 Detailed Description	776
4.22.2 Enumeration Type Documentation	776
4.22.3 Function Documentation	776
4.22.4 Variable Documentation	780

4.23 <code>std::rel_ops</code> Namespace Reference	783
4.23.1 Detailed Description	784
4.23.2 Function Documentation	784
4.24 <code>std::this_thread</code> Namespace Reference	785
4.24.1 Detailed Description	786
4.24.2 Function Documentation	786
4.25 <code>std::tr1</code> Namespace Reference	786
4.25.1 Detailed Description	789
4.26 <code>std::tr1::__detail</code> Namespace Reference	789
4.26.1 Detailed Description	789
4.27 <code>std::tr2</code> Namespace Reference	789
4.27.1 Detailed Description	790
4.28 <code>std::tr2::__detail</code> Namespace Reference	790
4.28.1 Detailed Description	790
5 Concept Documentation	791
5.1 <code>std::assignable_from</code> Concept Reference	791
5.1.1 Concept definition	791
5.1.2 Detailed Description	791
5.2 <code>std::common_reference_with</code> Concept Reference	791
5.2.1 Concept definition	791
5.2.2 Detailed Description	791
5.3 <code>std::common_with</code> Concept Reference	791
5.3.1 Concept definition	791
5.3.2 Detailed Description	791
5.4 <code>std::constructible_from</code> Concept Reference	791
5.4.1 Concept definition	791
5.4.2 Detailed Description	791
5.5 <code>std::convertible_to</code> Concept Reference	792
5.5.1 Concept definition	792
5.5.2 Detailed Description	792
5.6 <code>std::copy_constructible</code> Concept Reference	792
5.6.1 Concept definition	792
5.6.2 Detailed Description	792
5.7 <code>std::default_initializable</code> Concept Reference	792
5.7.1 Concept definition	792
5.7.2 Detailed Description	792
5.8 <code>std::derived_from</code> Concept Reference	792
5.8.1 Concept definition	792

5.8.2 Detailed Description	792
5.9 <code>std::destructible</code> Concept Reference	792
5.9.1 Concept definition	792
5.9.2 Detailed Description	792
5.10 <code>std::equivalence_relation</code> Concept Reference	793
5.10.1 Concept definition	793
5.10.2 Detailed Description	793
5.11 <code>std::indirectly_comparable</code> Concept Reference	793
5.11.1 Concept definition	793
5.11.2 Detailed Description	793
5.12 <code>std::indirectly_copyable</code> Concept Reference	793
5.12.1 Concept definition	793
5.12.2 Detailed Description	793
5.13 <code>std::indirectly_movable</code> Concept Reference	793
5.13.1 Concept definition	793
5.13.2 Detailed Description	793
5.14 <code>std::indirectly_readable</code> Concept Reference	793
5.14.1 Concept definition	793
5.14.2 Detailed Description	794
5.15 <code>std::indirectly_swappable</code> Concept Reference	794
5.15.1 Concept definition	794
5.15.2 Detailed Description	794
5.16 <code>std::indirectly_writable</code> Concept Reference	794
5.16.1 Concept definition	794
5.16.2 Detailed Description	794
5.17 <code>std::invocable</code> Concept Reference	794
5.17.1 Concept definition	794
5.17.2 Detailed Description	794
5.18 <code>std::mergeable</code> Concept Reference	794
5.18.1 Concept definition	795
5.18.2 Detailed Description	795
5.19 <code>std::move_constructible</code> Concept Reference	795
5.19.1 Concept definition	795
5.19.2 Detailed Description	795
5.20 <code>std::permutable</code> Concept Reference	795
5.20.1 Concept definition	795
5.20.2 Detailed Description	795
5.21 <code>std::predicate</code> Concept Reference	795
5.21.1 Concept definition	795

5.21.2 Detailed Description	795
5.22 <code>std::ranges::bidirectional_range</code> Concept Reference	795
5.22.1 Concept definition	795
5.22.2 Detailed Description	796
5.23 <code>std::ranges::borrowed_range</code> Concept Reference	796
5.23.1 Concept definition	796
5.23.2 Detailed Description	796
5.24 <code>std::ranges::common_range</code> Concept Reference	796
5.24.1 Concept definition	796
5.24.2 Detailed Description	796
5.25 <code>std::ranges::contiguous_range</code> Concept Reference	796
5.25.1 Concept definition	796
5.25.2 Detailed Description	796
5.26 <code>std::ranges::forward_range</code> Concept Reference	796
5.26.1 Concept definition	796
5.26.2 Detailed Description	796
5.27 <code>std::ranges::input_range</code> Concept Reference	797
5.27.1 Concept definition	797
5.27.2 Detailed Description	797
5.28 <code>std::ranges::output_range</code> Concept Reference	797
5.28.1 Concept definition	797
5.28.2 Detailed Description	797
5.29 <code>std::ranges::random_access_range</code> Concept Reference	797
5.29.1 Concept definition	797
5.29.2 Detailed Description	797
5.30 <code>std::ranges::range</code> Concept Reference	797
5.30.1 Concept definition	797
5.30.2 Detailed Description	797
5.31 <code>std::ranges::sized_range</code> Concept Reference	797
5.31.1 Concept definition	798
5.31.2 Detailed Description	798
5.32 <code>std::ranges::view</code> Concept Reference	798
5.32.1 Concept definition	798
5.32.2 Detailed Description	798
5.33 <code>std::ranges::viewable_range</code> Concept Reference	798
5.33.1 Concept definition	798
5.33.2 Detailed Description	798
5.34 <code>std::regular_invocable</code> Concept Reference	798
5.34.1 Concept definition	798

5.34.2 Detailed Description	798
5.35 <code>std::relation</code> Concept Reference	798
5.35.1 Concept definition	798
5.35.2 Detailed Description	799
5.36 <code>std::same_as</code> Concept Reference	799
5.36.1 Concept definition	799
5.36.2 Detailed Description	799
5.37 <code>std::sortable</code> Concept Reference	799
5.37.1 Concept definition	799
5.37.2 Detailed Description	799
5.38 <code>std::strict_weak_order</code> Concept Reference	799
5.38.1 Concept definition	799
5.38.2 Detailed Description	799
5.39 <code>std::uniform_random_bit_generator</code> Concept Reference	799
5.39.1 Concept definition	799
5.39.2 Detailed Description	799
5.40 <code>std::weakly_incrementable</code> Concept Reference	799
5.40.1 Concept definition	800
5.40.2 Detailed Description	800
6 Class Documentation	800
6.1 <code>__gnu_parallel::__accumulate_binop_reduct<_BinOp></code> Struct Template Reference	800
6.1.1 Detailed Description	800
6.2 <code>__gnu_parallel::__accumulate_selector<_It></code> Struct Template Reference	800
6.2.1 Detailed Description	801
6.2.2 Member Function Documentation	801
6.2.3 Member Data Documentation	802
6.3 <code>__gnu_parallel::__adjacent_difference_selector<_It></code> Struct Template Reference	802
6.3.1 Detailed Description	802
6.3.2 Member Data Documentation	802
6.4 <code>__gnu_parallel::__adjacent_find_selector</code> Struct Reference	803
6.4.1 Detailed Description	803
6.4.2 Member Function Documentation	803
6.5 <code>__gnu_cxx::__alloc_traits<_Alloc, typename></code> Struct Template Reference	804
6.5.1 Detailed Description	805
6.5.2 Member Typedef Documentation	806
6.5.3 Member Function Documentation	806
6.6 <code>std::__atomic_base<_ITp></code> Struct Template Reference	810
6.6.1 Detailed Description	812

6.7 std::__atomic_base< _PTp * > Struct Template Reference	812
6.7.1 Detailed Description	813
6.8 std::__atomic_flag_base Struct Reference	813
6.8.1 Detailed Description	813
6.9 std::__basic_future< _Res > Class Template Reference	814
6.9.1 Detailed Description	814
6.9.2 Member Function Documentation	815
6.10 __gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType > Class Template Reference	815
6.10.1 Detailed Description	816
6.10.2 Member Typedef Documentation	816
6.11 __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType > Class Template Reference	816
6.11.1 Detailed Description	817
6.11.2 Member Typedef Documentation	817
6.12 std::__codecvt_abstract_base< _InternT, _ExternT, _StateT > Class Template Reference	817
6.12.1 Detailed Description	818
6.12.2 Member Function Documentation	818
6.13 __gnu_cxx::__common_pool_policy< _PoolTp, _Thread > Struct Template Reference	821
6.13.1 Detailed Description	821
6.14 __gnu_parallel::__count_if_selector< _It, _Diff > Struct Template Reference	821
6.14.1 Detailed Description	822
6.14.2 Member Function Documentation	822
6.14.3 Member Data Documentation	823
6.15 __gnu_parallel::__count_selector< _It, _Diff > Struct Template Reference	823
6.15.1 Detailed Description	823
6.15.2 Member Function Documentation	823
6.15.3 Member Data Documentation	824
6.16 std::__ctype_abstract_base< _CharT > Class Template Reference	824
6.16.1 Detailed Description	826
6.16.2 Member Typedef Documentation	826
6.16.3 Member Function Documentation	826
6.17 std::filesystem::__directory_iterator_proxy Struct Reference	837
6.17.1 Detailed Description	837
6.18 std::tr2::__dynamic_bitset_base< _WordT, _Alloc > Struct Template Reference	838
6.18.1 Detailed Description	839
6.18.2 Member Data Documentation	839
6.19 __gnu_parallel::__fill_selector< _It > Struct Template Reference	840
6.19.1 Detailed Description	840
6.19.2 Member Function Documentation	840

6.19.3 Member Data Documentation	841
6.20 __gnu_parallel::__find_first_of_selector<_FIterator> Struct Template Reference	841
6.20.1 Detailed Description	841
6.20.2 Member Function Documentation	842
6.21 __gnu_parallel::__find_if_selector Struct Reference	842
6.21.1 Detailed Description	843
6.21.2 Member Function Documentation	843
6.22 __gnu_parallel::__for_each_selector<_It> Struct Template Reference	844
6.22.1 Detailed Description	844
6.22.2 Member Function Documentation	845
6.22.3 Member Data Documentation	845
6.23 __cxxabiv1::__forced_unwind Class Reference	845
6.23.1 Detailed Description	845
6.24 __gnu_parallel::__generate_selector<_It> Struct Template Reference	845
6.24.1 Detailed Description	846
6.24.2 Member Function Documentation	846
6.24.3 Member Data Documentation	847
6.25 __gnu_parallel::__generic_find_selector Struct Reference	847
6.25.1 Detailed Description	847
6.26 __gnu_parallel::__generic_for_each_selector<_It> Struct Template Reference	847
6.26.1 Detailed Description	849
6.26.2 Member Data Documentation	849
6.27 __gnu_parallel::__identity_selector<_It> Struct Template Reference	849
6.27.1 Detailed Description	849
6.27.2 Member Function Documentation	850
6.27.3 Member Data Documentation	850
6.28 __gnu_parallel::__inner_product_selector<_It, _It2, _Tp> Struct Template Reference	850
6.28.1 Detailed Description	851
6.28.2 Constructor & Destructor Documentation	851
6.28.3 Member Function Documentation	851
6.28.4 Member Data Documentation	852
6.29 std::__is_location_invariant<_Tp> Struct Template Reference	852
6.29.1 Detailed Description	852
6.30 std::__is_nullptr_t<_Tp> Struct Template Reference	853
6.30.1 Detailed Description	853
6.31 __gnu_parallel::__max_element_reduct<_Compare, _It> Struct Template Reference	853
6.31.1 Detailed Description	854
6.32 __gnu_parallel::__min_element_reduct<_Compare, _It> Struct Template Reference	854
6.32.1 Detailed Description	854

6.33	__gnu_cxx::__detail::__mini_vector< _Tp > Class Template Reference	854
6.33.1	Detailed Description	855
6.34	__gnu_parallel::__mismatch_selector Struct Reference	855
6.34.1	Detailed Description	856
6.34.2	Member Function Documentation	856
6.35	__gnu_cxx::__mt_alloc< _Tp, _Poolp > Class Template Reference	856
6.35.1	Detailed Description	858
6.36	__gnu_cxx::__mt_alloc_base< _Tp > Class Template Reference	858
6.36.1	Detailed Description	859
6.37	__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare > Struct Template Reference	859
6.37.1	Detailed Description	859
6.38	__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _↵ DifferenceTp, _Compare > Struct Template Reference	859
6.38.1	Detailed Description	859
6.39	__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare > Struct Template Reference	860
6.39.1	Detailed Description	860
6.40	__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _↵ DifferenceTp, _Compare > Struct Template Reference	860
6.40.1	Detailed Description	860
6.41	__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare > Struct Template Reference	860
6.41.1	Detailed Description	861
6.42	__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterlterator, _↵ RAIter3, _DifferenceTp, _Compare > Struct Template Reference	861
6.42.1	Detailed Description	861
6.43	std::__new_allocator< _Tp > Class Template Reference	861
6.43.1	Detailed Description	862
6.44	std::__numeric_limits_base Struct Reference	862
6.44.1	Detailed Description	863
6.44.2	Member Data Documentation	863
6.45	__gnu_cxx::__per_type_pool_policy< _Tp, _PoolTp, _Thread > Struct Template Reference	866
6.45.1	Detailed Description	866
6.46	__gnu_cxx::__pool< _Thread > Class Template Reference	866
6.46.1	Detailed Description	866
6.47	__gnu_cxx::__pool< false > Class Reference	866
6.47.1	Detailed Description	867
6.48	__gnu_cxx::__pool< true > Class Reference	868
6.48.1	Detailed Description	868
6.49	__gnu_cxx::__pool_alloc< _Tp > Class Template Reference	869

6.49.1 Detailed Description	869
6.50 <code>__gnu_cxx::__pool_alloc_base</code> Class Reference	870
6.50.1 Detailed Description	870
6.51 <code>__gnu_cxx::__pool_base</code> Struct Reference	871
6.51.1 Detailed Description	871
6.52 <code>__gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc></code> Class Template Reference	871
6.52.1 Detailed Description	873
6.53 <code>std::tr2::__reflection_typelist<_Elements></code> Struct Template Reference	873
6.53.1 Detailed Description	873
6.54 <code>std::tr2::__reflection_typelist<_First, _Rest...></code> Struct Template Reference	874
6.54.1 Detailed Description	874
6.55 <code>std::tr2::__reflection_typelist<></code> Struct Reference	874
6.55.1 Detailed Description	874
6.56 <code>__gnu_parallel::__replace_if_selector<_It, _Op, _Tp></code> Struct Template Reference	874
6.56.1 Detailed Description	875
6.56.2 Constructor & Destructor Documentation	875
6.56.3 Member Function Documentation	875
6.56.4 Member Data Documentation	875
6.57 <code>__gnu_parallel::__replace_selector<_It, _Tp></code> Struct Template Reference	876
6.57.1 Detailed Description	876
6.57.2 Constructor & Destructor Documentation	876
6.57.3 Member Function Documentation	877
6.57.4 Member Data Documentation	877
6.58 <code>__gnu_cxx::__scoped_lock</code> Class Reference	877
6.58.1 Detailed Description	877
6.59 <code>__gnu_parallel::__transform1_selector<_It></code> Struct Template Reference	878
6.59.1 Detailed Description	878
6.59.2 Member Function Documentation	878
6.59.3 Member Data Documentation	879
6.60 <code>__gnu_parallel::__transform2_selector<_It></code> Struct Template Reference	879
6.60.1 Detailed Description	879
6.60.2 Member Function Documentation	879
6.60.3 Member Data Documentation	880
6.61 <code>__gnu_parallel::__unary_negate<_Predicate, argument_type></code> Class Template Reference	880
6.61.1 Detailed Description	881
6.61.2 Member Typedef Documentation	881
6.62 <code>__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base></code> Class Template Reference	881
6.62.1 Detailed Description	884
6.62.2 Constructor & Destructor Documentation	884

6.62.3 Member Function Documentation	888
6.62.4 Member Data Documentation	933
6.63 <code>__gnu_debug::_After_nth_from<_Iterator></code> Class Template Reference	933
6.63.1 Detailed Description	933
6.64 <code>std::_Base_bitset<_Nw></code> Struct Template Reference	933
6.64.1 Detailed Description	934
6.64.2 Member Data Documentation	934
6.65 <code>std::_Base_bitset<0></code> Struct Reference	934
6.65.1 Detailed Description	935
6.66 <code>std::_Base_bitset<1></code> Struct Reference	935
6.66.1 Detailed Description	936
6.67 <code>__gnu_debug::_BeforeBeginHelper<_Sequence></code> Struct Template Reference	936
6.67.1 Detailed Description	936
6.68 <code>std::_Bind<_Signature></code> Class Template Reference	936
6.68.1 Detailed Description	936
6.69 <code>std::_Bind_result<_Result, _Signature></code> Class Template Reference	937
6.69.1 Detailed Description	937
6.70 <code>__gnu_cxx::__detail::_Bitmap_counter<_Tp></code> Class Template Reference	937
6.70.1 Detailed Description	937
6.71 <code>std::__detail::_BracketMatcher<_TraitsT, __icase, __collate></code> Struct Template Reference	937
6.71.1 Detailed Description	938
6.72 <code>__gnu_cxx::_Caster<_ToType></code> Struct Template Reference	938
6.72.1 Detailed Description	938
6.73 <code>__gnu_cxx::_Char_types<_CharT></code> Struct Template Reference	938
6.73.1 Detailed Description	939
6.74 <code>__gnu_pbds::detail::pat_trie_base::_Clter<Node, Leaf, Head, Inode, Is_Forward_Iterator></code> Class Template Reference	939
6.74.1 Detailed Description	940
6.75 <code>std::__detail::_Compiler<_TraitsT></code> Class Template Reference	940
6.75.1 Detailed Description	941
6.76 <code>std::__parallel::_CRandNumber<_MustBeInt></code> Struct Template Reference	941
6.76.1 Detailed Description	941
6.77 <code>std::_Deque_base<_Tp, _Alloc></code> Class Template Reference	941
6.77.1 Detailed Description	942
6.77.2 Member Function Documentation	943
6.78 <code>std::_Deque_iterator<_Tp, _Ref, _Ptr></code> Struct Template Reference	944
6.78.1 Detailed Description	945
6.78.2 Member Function Documentation	945
6.79 <code>__gnu_parallel::_DRandomShufflingGlobalData<_RAIter></code> Struct Template Reference	946

6.79.1 Detailed Description	946
6.79.2 Constructor & Destructor Documentation	946
6.79.3 Member Data Documentation	946
6.80 __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator > Struct Template Reference	947
6.80.1 Detailed Description	948
6.80.2 Member Data Documentation	948
6.81 __gnu_parallel::_DummyReduct Struct Reference	948
6.81.1 Detailed Description	949
6.82 __gnu_debug::_Equal_to<_Type > Class Template Reference	949
6.82.1 Detailed Description	949
6.83 __gnu_parallel::_EqualFromLess<_T1, _T2, _Compare > Class Template Reference	949
6.83.1 Detailed Description	950
6.83.2 Member Typedef Documentation	950
6.84 __gnu_parallel::_EqualTo<_T1, _T2 > Struct Template Reference	950
6.84.1 Detailed Description	951
6.84.2 Member Typedef Documentation	951
6.85 std::__detail::_Executor<_Biliter, _Alloc, _TraitsT, __dfs_mode > Class Template Reference	951
6.85.1 Detailed Description	952
6.86 __gnu_cxx::_ExtPtr_allocator<_Tp > Class Template Reference	952
6.86.1 Detailed Description	953
6.87 __gnu_cxx::__detail::_Ffit_finder<_Tp > Class Template Reference	953
6.87.1 Detailed Description	953
6.88 std::_Function_base Class Reference	954
6.88.1 Detailed Description	954
6.89 std::_Fwd_list_base<_Tp, _Alloc > Struct Template Reference	954
6.89.1 Detailed Description	956
6.90 std::_Fwd_list_const_iterator<_Tp > Struct Template Reference	956
6.90.1 Detailed Description	956
6.90.2 Friends And Related Symbol Documentation	957
6.91 std::_Fwd_list_iterator<_Tp > Struct Template Reference	957
6.91.1 Detailed Description	957
6.91.2 Friends And Related Symbol Documentation	958
6.92 std::_Fwd_list_node<_Tp > Struct Template Reference	958
6.92.1 Detailed Description	958
6.93 std::_Fwd_list_node_base Struct Reference	959
6.93.1 Detailed Description	959
6.94 __gnu_parallel::_GuardedIterator<_RAIter, _Compare > Class Template Reference	959
6.94.1 Detailed Description	960
6.94.2 Constructor & Destructor Documentation	960

6.94.3 Member Function Documentation	960
6.94.4 Friends And Related Symbol Documentation	961
6.95 <code>__gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata></code> Struct Template Reference	961
6.95.1 Detailed Description	962
6.96 <code>__gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata></code> Struct Template Reference	962
6.96.1 Detailed Description	964
6.97 <code>__gnu_cxx::_Invalid_type</code> Struct Reference	964
6.97.1 Detailed Description	964
6.98 <code>__gnu_pbds::detail::pat_trie_base::_Iter<Node, Leaf, Head, Inode, Is_Forward_Iterator></code> Class Template Reference	964
6.98.1 Detailed Description	966
6.99 <code>__gnu_parallel::_IteratorPair<_Iterator1, _Iterator2, _IteratorCategory></code> Class Template Reference	966
6.99.1 Detailed Description	967
6.99.2 Member Typedef Documentation	967
6.99.3 Member Function Documentation	967
6.99.4 Friends And Related Symbol Documentation	968
6.99.5 Member Data Documentation	969
6.100 <code>__gnu_parallel::_IteratorTriple<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory></code> Class Template Reference	969
6.100.1 Detailed Description	970
6.101 <code>__gnu_parallel::_Job<_DifferenceTp></code> Struct Template Reference	970
6.101.1 Detailed Description	970
6.101.2 Member Data Documentation	970
6.102 <code>__gnu_pbds::detail::pat_trie_base::_Leaf<_ATraits, Metadata></code> Struct Template Reference	971
6.102.1 Detailed Description	972
6.103 <code>__gnu_parallel::_Less<_T1, _T2></code> Struct Template Reference	972
6.103.1 Detailed Description	973
6.103.2 Member Typedef Documentation	973
6.104 <code>__gnu_parallel::_Lexicographic<_T1, _T2, _Compare></code> Class Template Reference	973
6.104.1 Detailed Description	974
6.104.2 Member Typedef Documentation	974
6.105 <code>__gnu_parallel::_LexicographicReverse<_T1, _T2, _Compare></code> Class Template Reference	974
6.105.1 Detailed Description	975
6.105.2 Member Typedef Documentation	975
6.106 <code>std::_List_base<_Tp, _Alloc></code> Class Template Reference	975
6.106.1 Detailed Description	977
6.107 <code>std::_List_const_iterator<_Tp></code> Struct Template Reference	977
6.107.1 Detailed Description	978
6.108 <code>std::_List_iterator<_Tp></code> Struct Template Reference	978
6.108.1 Detailed Description	978

6.109	std::_List_node< _Tp > Struct Template Reference	979
6.109.1	Detailed Description	979
6.110	std::_detail::_List_node_base Struct Reference	979
6.110.1	Detailed Description	980
6.111	std::_detail::_List_node_header Struct Reference	980
6.111.1	Detailed Description	981
6.112	__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser Struct Reference	981
6.112.1	Detailed Description	982
6.112.2	Member Data Documentation	982
6.113	__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser Struct Reference	982
6.113.1	Detailed Description	982
6.114	__gnu_parallel::_LoserTree< __stable, _Tp, _Compare > Class Template Reference	983
6.114.1	Detailed Description	983
6.114.2	Member Function Documentation	983
6.114.3	Member Data Documentation	984
6.115	__gnu_parallel::_LoserTree< false, _Tp, _Compare > Class Template Reference	984
6.115.1	Detailed Description	985
6.115.2	Member Function Documentation	985
6.116	__gnu_parallel::_LoserTreeBase< _Tp, _Compare > Class Template Reference	986
6.116.1	Detailed Description	987
6.116.2	Constructor & Destructor Documentation	988
6.116.3	Member Function Documentation	988
6.116.4	Member Data Documentation	989
6.117	__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare > Class Template Reference	989
6.117.1	Detailed Description	990
6.118	__gnu_parallel::_LoserTreePointer< false, _Tp, _Compare > Class Template Reference	990
6.118.1	Detailed Description	991
6.119	__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare > Class Template Reference	991
6.119.1	Detailed Description	992
6.120	__gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare > Class Template Reference	992
6.120.1	Detailed Description	992
6.121	__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare > Class Template Reference	993
6.121.1	Detailed Description	993
6.122	__gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare > Class Template Reference	994
6.122.1	Detailed Description	994
6.123	__gnu_parallel::_LoserTreeTraits< _Tp > Struct Template Reference	994
6.123.1	Detailed Description	995
6.123.2	Member Data Documentation	995
6.124	__gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare > Class Template Reference	995

6.124.1 Detailed Description	996
6.125 <code>__gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare ></code> Class Template Reference	996
6.125.1 Detailed Description	997
6.126 <code>__gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare ></code> Class Template Reference	997
6.126.1 Detailed Description	997
6.127 <code>__gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc ></code> Struct Template Reference	998
6.127.1 Detailed Description	998
6.128 <code>__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc ></code> Struct Template Reference	998
6.128.1 Detailed Description	998
6.129 <code>__gnu_parallel::_Multiplies< _Tp1, _Tp2, _Result ></code> Struct Template Reference	999
6.129.1 Detailed Description	999
6.129.2 Member Typedef Documentation	999
6.130 <code>__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata ></code> Struct Template Reference	1000
6.130.1 Detailed Description	1001
6.131 <code>__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc ></code> Class Template Reference	1001
6.131.1 Detailed Description	1002
6.131.2 Member Typedef Documentation	1002
6.131.3 Member Function Documentation	1002
6.132 <code>std::_Node_handle< _Key, _Value, _NodeAlloc ></code> Class Template Reference	1003
6.132.1 Detailed Description	1004
6.133 <code>std::_Node_handle< _Value, _Value, _NodeAlloc ></code> Class Template Reference	1005
6.133.1 Detailed Description	1005
6.134 <code>std::_Node_handle_common< _Val, _NodeAlloc ></code> Class Template Reference	1006
6.134.1 Detailed Description	1006
6.135 <code>std::_Node_insert_return< _Iterator, _NodeHandle ></code> Struct Template Reference	1006
6.135.1 Detailed Description	1006
6.136 <code>__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc ></code> Class Template Reference	1006
6.136.1 Detailed Description	1008
6.136.2 Member Typedef Documentation	1008
6.136.3 Member Function Documentation	1008
6.137 <code>__gnu_debug::_Not_equal_to< _Type ></code> Class Template Reference	1009
6.137.1 Detailed Description	1009
6.138 <code>std::_Not_fn< _Fn ></code> Class Template Reference	1010
6.138.1 Detailed Description	1010
6.139 <code>__gnu_parallel::_Nothing</code> Struct Reference	1010
6.139.1 Detailed Description	1010
6.139.2 Member Function Documentation	1011
6.140 <code>std::_Optional_base< _Tp, bool, bool ></code> Struct Template Reference	1011

6.140.1 Detailed Description	1012
6.141 __gnu_parallel::_Piece< _DifferenceTp > Struct Template Reference	1012
6.141.1 Detailed Description	1012
6.141.2 Member Data Documentation	1013
6.142 std::_Placeholder< _Num > Struct Template Reference	1013
6.142.1 Detailed Description	1013
6.143 __gnu_parallel::_Plus< _Tp1, _Tp2, _Result > Struct Template Reference	1013
6.143.1 Detailed Description	1014
6.143.2 Member Typedef Documentation	1014
6.144 __gnu_parallel::_PMWMSSortingData< _RAIter > Struct Template Reference	1014
6.144.1 Detailed Description	1015
6.144.2 Member Data Documentation	1015
6.145 __gnu_cxx::_Pointer_adapter< _Storage_policy > Class Template Reference	1016
6.145.1 Detailed Description	1018
6.146 __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp > Class Template Reference	1018
6.146.1 Detailed Description	1018
6.146.2 Constructor & Destructor Documentation	1019
6.146.3 Member Function Documentation	1019
6.147 __gnu_parallel::_PseudoSequenceIterator< _Tp, _DifferenceTp > Class Template Reference	1019
6.147.1 Detailed Description	1020
6.148 __gnu_parallel::_QSBThreadLocal< _RAIter > Struct Template Reference	1020
6.148.1 Detailed Description	1020
6.148.2 Member Typedef Documentation	1020
6.148.3 Constructor & Destructor Documentation	1021
6.148.4 Member Data Documentation	1021
6.149 std::_detail::_Quoted_string< _String, _CharT > Struct Template Reference	1021
6.149.1 Detailed Description	1022
6.150 __gnu_parallel::_RandomNumber Class Reference	1022
6.150.1 Detailed Description	1022
6.150.2 Constructor & Destructor Documentation	1022
6.150.3 Member Function Documentation	1023
6.151 __gnu_cxx::_Relative_pointer_impl< _Tp > Class Template Reference	1023
6.151.1 Detailed Description	1023
6.152 __gnu_cxx::_Relative_pointer_impl< const _Tp > Class Template Reference	1024
6.152.1 Detailed Description	1024
6.153 __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp > Class Template Reference	1024
6.153.1 Detailed Description	1024
6.153.2 Constructor & Destructor Documentation	1025
6.153.3 Member Function Documentation	1025

6.154	__gnu_debug::_Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >	
	Class Template Reference	1025
6.154.1	Detailed Description	1027
6.155	__gnu_debug::_Safe_forward_list< _SafeSequence >	Class Template Reference 1027
6.155.1	Detailed Description	1028
6.155.2	Member Function Documentation	1028
6.155.3	Member Data Documentation	1029
6.156	__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >	Class Template Reference 1029
6.156.1	Detailed Description	1032
6.156.2	Constructor & Destructor Documentation	1032
6.156.3	Member Function Documentation	1033
6.156.4	Member Data Documentation	1038
6.157	__gnu_debug::_Safe_iterator_base	Class Reference 1039
6.157.1	Detailed Description	1040
6.157.2	Constructor & Destructor Documentation	1040
6.157.3	Member Function Documentation	1041
6.157.4	Member Data Documentation	1042
6.158	__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >	Class Template Reference 1043
6.158.1	Detailed Description	1045
6.158.2	Constructor & Destructor Documentation	1045
6.158.3	Member Function Documentation	1046
6.158.4	Member Data Documentation	1051
6.159	__gnu_debug::_Safe_local_iterator_base	Class Reference 1051
6.159.1	Detailed Description	1053
6.159.2	Constructor & Destructor Documentation	1053
6.159.3	Member Function Documentation	1053
6.159.4	Member Data Documentation	1055
6.160	__gnu_debug::_Safe_node_sequence< _Sequence >	Class Template Reference 1055
6.160.1	Detailed Description	1056
6.160.2	Member Function Documentation	1057
6.160.3	Member Data Documentation	1058
6.161	__gnu_debug::_Safe_sequence< _Sequence >	Class Template Reference 1058
6.161.1	Detailed Description	1059
6.161.2	Member Function Documentation	1060
6.161.3	Member Data Documentation	1061
6.162	__gnu_debug::_Safe_sequence_base	Class Reference 1061
6.162.1	Detailed Description	1062
6.162.2	Constructor & Destructor Documentation	1062
6.162.3	Member Function Documentation	1062

6.162.4 Member Data Documentation	1063
6.163 <code>__gnu_debug::_Safe_unordered_container<_Container></code> Class Template Reference	1063
6.163.1 Detailed Description	1065
6.163.2 Member Function Documentation	1065
6.163.3 Member Data Documentation	1066
6.164 <code>__gnu_debug::_Safe_unordered_container_base</code> Class Reference	1067
6.164.1 Detailed Description	1068
6.164.2 Constructor & Destructor Documentation	1068
6.164.3 Member Function Documentation	1068
6.164.4 Member Data Documentation	1069
6.165 <code>__gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence></code> Class Template Reference	1070
6.165.1 Detailed Description	1070
6.166 <code>__gnu_parallel::SamplingSorter<__stable, _RAIter, _StrictWeakOrdering></code> Struct Template Reference	1070
6.166.1 Detailed Description	1071
6.167 <code>__gnu_parallel::SamplingSorter<false, _RAIter, _StrictWeakOrdering></code> Struct Template Reference	1071
6.167.1 Detailed Description	1071
6.168 <code>std::__detail::Scanner<_CharT></code> Class Template Reference	1071
6.168.1 Detailed Description	1072
6.168.2 Member Enumeration Documentation	1072
6.169 <code>__gnu_debug::Sequence_traits<_Sequence></code> Struct Template Reference	1073
6.169.1 Detailed Description	1073
6.170 <code>__gnu_parallel::Settings</code> Struct Reference	1073
6.170.1 Detailed Description	1074
6.170.2 Member Function Documentation	1074
6.170.3 Member Data Documentation	1075
6.171 <code>std::Sp_ebo_helper<_Nm, _Tp, false></code> Struct Template Reference	1079
6.171.1 Detailed Description	1079
6.172 <code>std::Sp_ebo_helper<_Nm, _Tp, true></code> Struct Template Reference	1079
6.172.1 Detailed Description	1080
6.173 <code>__gnu_parallel::SplitConsistently<__exact, _RAIter, _Compare, _SortingPlacesIterator></code> Struct Template Reference	1080
6.173.1 Detailed Description	1080
6.174 <code>__gnu_parallel::SplitConsistently<false, _RAIter, _Compare, _SortingPlacesIterator></code> Struct Template Reference	1080
6.174.1 Detailed Description	1080
6.175 <code>__gnu_parallel::SplitConsistently<true, _RAIter, _Compare, _SortingPlacesIterator></code> Struct Template Reference	1080
6.175.1 Detailed Description	1081
6.176 <code>std::__detail::StateSeq<_TraitsT></code> Class Template Reference	1081
6.176.1 Detailed Description	1081

6.177 <code>__gnu_cxx::Std_pointer_impl<_Tp></code> Class Template Reference	1081
6.177.1 Detailed Description	1082
6.178 <code>std::Temporary_buffer<_ForwardIterator, _Tp></code> Class Template Reference	1082
6.178.1 Detailed Description	1083
6.178.2 Constructor & Destructor Documentation	1083
6.178.3 Member Function Documentation	1083
6.179 <code>std::Tuple_impl<_Idx, _Elements></code> Struct Template Reference	1083
6.179.1 Detailed Description	1083
6.180 <code>std::Tuple_impl<_Idx, _Head, _Tail...></code> Struct Template Reference	1084
6.180.1 Detailed Description	1085
6.181 <code>__gnu_cxx::Unqualified_type<_Tp></code> Struct Template Reference	1085
6.181.1 Detailed Description	1085
6.182 <code>std::_Vector_base<_Tp, _Alloc></code> Struct Template Reference	1085
6.182.1 Detailed Description	1087
6.183 <code>std::add_const<_Tp></code> Struct Template Reference	1087
6.183.1 Detailed Description	1087
6.184 <code>std::add_cv<_Tp></code> Struct Template Reference	1087
6.184.1 Detailed Description	1088
6.185 <code>std::add_lvalue_reference<_Tp></code> Struct Template Reference	1088
6.185.1 Detailed Description	1088
6.186 <code>std::add_pointer<_Tp></code> Struct Template Reference	1088
6.186.1 Detailed Description	1088
6.187 <code>std::add_rvalue_reference<_Tp></code> Struct Template Reference	1088
6.187.1 Detailed Description	1089
6.188 <code>std::add_volatile<_Tp></code> Struct Template Reference	1089
6.188.1 Detailed Description	1089
6.189 <code>std::adopt_lock_t</code> Struct Reference	1089
6.189.1 Detailed Description	1089
6.190 <code>std::aligned_storage<_Len, _Align></code> Struct Template Reference	1089
6.190.1 Detailed Description	1089
6.191 <code>std::aligned_union<_Len, _Types></code> Struct Template Reference	1089
6.191.1 Detailed Description	1090
6.191.2 Member Typedef Documentation	1090
6.192 <code>std::alignment_of<_Tp></code> Struct Template Reference	1090
6.192.1 Detailed Description	1091
6.193 <code>std::allocator<_Tp></code> Class Template Reference	1091
6.193.1 Detailed Description	1092
6.194 <code>std::allocator<void></code> Class Reference	1092
6.194.1 Detailed Description	1092

6.195 std::allocator_traits< _Alloc > Struct Template Reference	1092
6.195.1 Detailed Description	1094
6.195.2 Member Typedef Documentation	1094
6.195.3 Member Function Documentation	1095
6.196 std::allocator_traits< allocator< _Tp > > Struct Template Reference	1098
6.196.1 Detailed Description	1099
6.196.2 Member Typedef Documentation	1099
6.196.3 Member Function Documentation	1100
6.197 std::allocator_traits< allocator< void > > Struct Reference	1102
6.197.1 Detailed Description	1103
6.197.2 Member Typedef Documentation	1103
6.197.3 Member Function Documentation	1104
6.198 std::allocator_traits< pmr::polymorphic_allocator< _Tp > > Struct Template Reference	1106
6.198.1 Detailed Description	1106
6.198.2 Member Typedef Documentation	1107
6.198.3 Member Function Documentation	1108
6.199 __gnu_cxx::limit_condition::always_adjustor Struct Reference	1110
6.199.1 Detailed Description	1110
6.200 __gnu_cxx::random_condition::always_adjustor Struct Reference	1110
6.200.1 Detailed Description	1111
6.201 __gnu_cxx::annotate_base Struct Reference	1111
6.201.1 Detailed Description	1111
6.202 std::any Class Reference	1112
6.202.1 Detailed Description	1112
6.202.2 Constructor & Destructor Documentation	1112
6.202.3 Member Function Documentation	1113
6.203 std::experimental::fundamentals_v1::any Class Reference	1115
6.203.1 Detailed Description	1115
6.203.2 Constructor & Destructor Documentation	1115
6.203.3 Member Function Documentation	1116
6.204 std::array< _Tp, _Nm > Struct Template Reference	1117
6.204.1 Detailed Description	1118
6.205 __gnu_pbds::associative_tag Struct Reference	1118
6.205.1 Detailed Description	1119
6.206 std::atomic< _Tp > Class Template Reference	1119
6.206.1 Detailed Description	1119
6.207 std::atomic< _Tp * > Struct Template Reference	1120
6.207.1 Detailed Description	1121
6.208 std::atomic< bool > Struct Reference	1121

6.208.1 Detailed Description	1122
6.209 std::atomic< char > Struct Reference	1122
6.209.1 Detailed Description	1124
6.210 std::atomic< char16_t > Struct Reference	1124
6.210.1 Detailed Description	1126
6.211 std::atomic< char32_t > Struct Reference	1126
6.211.1 Detailed Description	1128
6.212 std::atomic< int > Struct Reference	1128
6.212.1 Detailed Description	1130
6.213 std::atomic< long > Struct Reference	1130
6.213.1 Detailed Description	1132
6.214 std::atomic< long long > Struct Reference	1132
6.214.1 Detailed Description	1134
6.215 std::atomic< short > Struct Reference	1134
6.215.1 Detailed Description	1136
6.216 std::atomic< signed char > Struct Reference	1136
6.216.1 Detailed Description	1138
6.217 std::atomic< unsigned char > Struct Reference	1138
6.217.1 Detailed Description	1140
6.218 std::atomic< unsigned int > Struct Reference	1140
6.218.1 Detailed Description	1142
6.219 std::atomic< unsigned long > Struct Reference	1142
6.219.1 Detailed Description	1144
6.220 std::atomic< unsigned long long > Struct Reference	1144
6.220.1 Detailed Description	1146
6.221 std::atomic< unsigned short > Struct Reference	1146
6.221.1 Detailed Description	1148
6.222 std::atomic< wchar_t > Struct Reference	1148
6.222.1 Detailed Description	1150
6.223 std::atomic_flag Struct Reference	1150
6.223.1 Detailed Description	1151
6.224 std::atomic_ref< _Tp > Struct Template Reference	1151
6.224.1 Detailed Description	1151
6.225 std::auto_ptr< _Tp > Class Template Reference	1151
6.225.1 Detailed Description	1152
6.225.2 Member Typedef Documentation	1152
6.225.3 Constructor & Destructor Documentation	1152
6.225.4 Member Function Documentation	1154
6.226 std::auto_ptr_ref< _Tp1 > Struct Template Reference	1155

6.226.1 Detailed Description	1156
6.227 std::back_insert_iterator< _Container > Class Template Reference	1156
6.227.1 Detailed Description	1157
6.227.2 Member Typedef Documentation	1157
6.227.3 Constructor & Destructor Documentation	1157
6.227.4 Member Function Documentation	1157
6.228 std::bad_alloc Class Reference	1158
6.228.1 Detailed Description	1159
6.228.2 Member Function Documentation	1159
6.229 std::bad_any_cast Class Reference	1159
6.229.1 Detailed Description	1159
6.229.2 Member Function Documentation	1160
6.230 std::experimental::fundamentals_v1::bad_any_cast Class Reference	1160
6.230.1 Detailed Description	1160
6.230.2 Member Function Documentation	1160
6.231 std::bad_cast Class Reference	1161
6.231.1 Detailed Description	1161
6.231.2 Member Function Documentation	1161
6.232 std::bad_exception Class Reference	1162
6.232.1 Detailed Description	1162
6.232.2 Member Function Documentation	1162
6.233 std::bad_function_call Class Reference	1162
6.233.1 Detailed Description	1163
6.233.2 Member Function Documentation	1163
6.234 std::bad_optional_access Class Reference	1163
6.234.1 Detailed Description	1164
6.234.2 Member Function Documentation	1164
6.235 std::experimental::fundamentals_v1::bad_optional_access Class Reference	1164
6.235.1 Detailed Description	1164
6.235.2 Member Function Documentation	1165
6.236 std::bad_typeid Class Reference	1165
6.236.1 Detailed Description	1165
6.236.2 Member Function Documentation	1165
6.237 std::bad_weak_ptr Class Reference	1165
6.237.1 Detailed Description	1166
6.237.2 Member Function Documentation	1166
6.238 __gnu_parallel::balanced_quicksort_tag Struct Reference	1166
6.238.1 Detailed Description	1167
6.238.2 Member Function Documentation	1167

6.239 __gnu_parallel::balanced_tag Struct Reference	1167
6.239.1 Detailed Description	1168
6.239.2 Member Function Documentation	1168
6.240 std::tr2::bases< _Tp > Struct Template Reference	1168
6.240.1 Detailed Description	1168
6.241 __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc > Class Template Reference	1168
6.241.1 Detailed Description	1169
6.242 __gnu_pbds::basic_branch_tag Struct Reference	1170
6.242.1 Detailed Description	1170
6.243 std::basic_filebuf< _CharT, _Traits > Class Template Reference	1170
6.243.1 Detailed Description	1173
6.243.2 Constructor & Destructor Documentation	1173
6.243.3 Member Function Documentation	1174
6.243.4 Member Data Documentation	1188
6.244 std::basic_fstream< _CharT, _Traits > Class Template Reference	1191
6.244.1 Detailed Description	1197
6.244.2 Member Typedef Documentation	1197
6.244.3 Member Enumeration Documentation	1199
6.244.4 Constructor & Destructor Documentation	1199
6.244.5 Member Function Documentation	1200
6.244.6 Member Data Documentation	1235
6.245 __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc > Class Template Reference	1240
6.245.1 Detailed Description	1241
6.246 __gnu_pbds::basic_hash_tag Struct Reference	1241
6.246.1 Detailed Description	1242
6.247 std::basic_ifstream< _CharT, _Traits > Class Template Reference	1243
6.247.1 Detailed Description	1248
6.247.2 Member Typedef Documentation	1248
6.247.3 Member Enumeration Documentation	1250
6.247.4 Constructor & Destructor Documentation	1250
6.247.5 Member Function Documentation	1251
6.247.6 Member Data Documentation	1278
6.248 __gnu_pbds::basic_invalidation_guarantee Struct Reference	1283
6.248.1 Detailed Description	1284
6.249 std::basic_ios< _CharT, _Traits > Class Template Reference	1284
6.249.1 Detailed Description	1287
6.249.2 Member Typedef Documentation	1287
6.249.3 Member Enumeration Documentation	1290

6.249.4 Constructor & Destructor Documentation	1290
6.249.5 Member Function Documentation	1291
6.249.6 Member Data Documentation	1302
6.250 std::basic_istream< _CharT, _Traits > Class Template Reference	1306
6.250.1 Detailed Description	1312
6.250.2 Member Typedef Documentation	1312
6.250.3 Member Enumeration Documentation	1314
6.250.4 Constructor & Destructor Documentation	1314
6.250.5 Member Function Documentation	1315
6.250.6 Member Data Documentation	1348
6.251 std::basic_istream< _CharT, _Traits > Class Template Reference	1352
6.251.1 Detailed Description	1357
6.251.2 Member Typedef Documentation	1357
6.251.3 Member Enumeration Documentation	1359
6.251.4 Constructor & Destructor Documentation	1359
6.251.5 Member Function Documentation	1360
6.251.6 Member Data Documentation	1385
6.252 std::basic_istream< _CharT, _Traits, _Alloc > Class Template Reference	1390
6.252.1 Detailed Description	1395
6.252.2 Member Typedef Documentation	1395
6.252.3 Member Enumeration Documentation	1397
6.252.4 Constructor & Destructor Documentation	1397
6.252.5 Member Function Documentation	1398
6.252.6 Member Data Documentation	1426
6.253 std::basic_ofstream< _CharT, _Traits > Class Template Reference	1432
6.253.1 Detailed Description	1436
6.253.2 Member Typedef Documentation	1436
6.253.3 Member Enumeration Documentation	1438
6.253.4 Constructor & Destructor Documentation	1439
6.253.5 Member Function Documentation	1440
6.253.6 Member Data Documentation	1461
6.254 std::basic_ostream< _CharT, _Traits > Class Template Reference	1465
6.254.1 Detailed Description	1469
6.254.2 Member Typedef Documentation	1470
6.254.3 Member Enumeration Documentation	1471
6.254.4 Constructor & Destructor Documentation	1472
6.254.5 Member Function Documentation	1472
6.254.6 Member Data Documentation	1493
6.255 std::basic_ostream< _CharT, _Traits, _Alloc > Class Template Reference	1497

6.255.1 Detailed Description	1501
6.255.2 Member Typedef Documentation	1502
6.255.3 Member Enumeration Documentation	1504
6.255.4 Constructor & Destructor Documentation	1504
6.255.5 Member Function Documentation	1505
6.255.6 Member Data Documentation	1525
6.256 std::basic_regex< _Ch_type, _Rx_traits > Class Template Reference	1529
6.256.1 Detailed Description	1531
6.256.2 Constructor & Destructor Documentation	1531
6.256.3 Member Function Documentation	1534
6.257 std::basic_streambuf< _CharT, _Traits > Class Template Reference	1539
6.257.1 Detailed Description	1541
6.257.2 Member Typedef Documentation	1542
6.257.3 Constructor & Destructor Documentation	1543
6.257.4 Member Function Documentation	1543
6.257.5 Member Data Documentation	1555
6.258 __gnu_debug::basic_string< _CharT, _Traits, _Allocator > Class Template Reference	1556
6.258.1 Detailed Description	1561
6.258.2 Member Function Documentation	1561
6.258.3 Member Data Documentation	1590
6.259 std::basic_string< _CharT, _Traits, _Alloc > Class Template Reference	1590
6.259.1 Detailed Description	1595
6.259.2 Constructor & Destructor Documentation	1596
6.259.3 Member Function Documentation	1601
6.259.4 Member Data Documentation	1656
6.260 std::basic_string_view< _CharT, _Traits > Class Template Reference	1656
6.260.1 Detailed Description	1658
6.261 std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits > Class Template Reference	1658
6.261.1 Detailed Description	1660
6.262 std::basic_stringbuf< _CharT, _Traits, _Alloc > Class Template Reference	1661
6.262.1 Detailed Description	1663
6.262.2 Constructor & Destructor Documentation	1663
6.262.3 Member Function Documentation	1664
6.262.4 Member Data Documentation	1677
6.263 std::basic_stringstream< _CharT, _Traits, _Alloc > Class Template Reference	1678
6.263.1 Detailed Description	1684
6.263.2 Member Typedef Documentation	1684
6.263.3 Member Enumeration Documentation	1686
6.263.4 Constructor & Destructor Documentation	1687

6.263.5 Member Function Documentation	1687
6.263.6 Member Data Documentation	1721
6.264 std::bernoulli_distribution Class Reference	1725
6.264.1 Detailed Description	1726
6.264.2 Member Typedef Documentation	1726
6.264.3 Constructor & Destructor Documentation	1726
6.264.4 Member Function Documentation	1726
6.264.5 Friends And Related Symbol Documentation	1727
6.265 std::bidirectional_iterator_tag Struct Reference	1728
6.265.1 Detailed Description	1728
6.266 __gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > Class Template Reference	1728
6.266.1 Detailed Description	1730
6.267 __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > Class Template Reference	1730
6.267.1 Detailed Description	1731
6.267.2 Member Typedef Documentation	1731
6.267.3 Member Function Documentation	1732
6.268 __gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > Class Template Reference	1733
6.268.1 Detailed Description	1734
6.269 __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc > Class Template Reference	1734
6.269.1 Detailed Description	1735
6.269.2 Member Typedef Documentation	1735
6.269.3 Member Function Documentation	1736
6.270 __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference	1737
6.270.1 Detailed Description	1738
6.270.2 Member Typedef Documentation	1738
6.271 __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference	1738
6.271.1 Detailed Description	1739
6.271.2 Member Typedef Documentation	1739
6.272 __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 > Class Template Reference	1739
6.272.1 Detailed Description	1740
6.272.2 Member Typedef Documentation	1740
6.273 std::binary_function< _Arg1, _Arg2, _Result > Struct Template Reference	1740
6.273.1 Detailed Description	1741
6.273.2 Member Typedef Documentation	1741
6.274 __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1742

6.274.1 Detailed Description	1744
6.275 __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc > Class Template Reference	1744
6.275.1 Detailed Description	1745
6.275.2 Member Typedef Documentation	1745
6.275.3 Constructor & Destructor Documentation	1746
6.275.4 Member Function Documentation	1746
6.276 __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc > Class Template Reference	1747
6.276.1 Detailed Description	1748
6.276.2 Member Typedef Documentation	1748
6.276.3 Constructor & Destructor Documentation	1749
6.276.4 Member Function Documentation	1749
6.277 __gnu_pbds::binary_heap_tag Struct Reference	1750
6.277.1 Detailed Description	1750
6.278 std::binary_negate< _Predicate > Class Template Reference	1750
6.278.1 Detailed Description	1751
6.278.2 Member Typedef Documentation	1751
6.279 std::binder1st< _Operation > Class Template Reference	1752
6.279.1 Detailed Description	1752
6.279.2 Member Typedef Documentation	1752
6.280 std::binder2nd< _Operation > Class Template Reference	1753
6.280.1 Detailed Description	1753
6.280.2 Member Typedef Documentation	1754
6.281 std::binomial_distribution< _IntType > Class Template Reference	1754
6.281.1 Detailed Description	1755
6.281.2 Member Typedef Documentation	1755
6.281.3 Member Function Documentation	1755
6.281.4 Friends And Related Symbol Documentation	1756
6.282 __gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1757
6.282.1 Detailed Description	1759
6.283 __gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1759
6.283.1 Detailed Description	1761
6.284 __gnu_pbds::binomial_heap_tag Struct Reference	1761
6.284.1 Detailed Description	1762
6.285 __gnu_cxx::bitmap_allocator< _Tp > Class Template Reference	1762
6.285.1 Detailed Description	1763
6.285.2 Member Function Documentation	1763
6.286 std::__debug::bitset< _Nb > Class Template Reference	1764
6.286.1 Detailed Description	1765

6.287	std::bitset< _Nb > Class Template Reference	1765
6.287.1	Detailed Description	1767
6.287.2	Constructor & Destructor Documentation	1768
6.287.3	Member Function Documentation	1769
6.288	std::tr2::bool_set Class Reference	1775
6.288.1	Detailed Description	1776
6.288.2	Constructor & Destructor Documentation	1776
6.288.3	Member Function Documentation	1776
6.289	__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc > Struct Template Reference	1777
6.289.1	Detailed Description	1777
6.290	__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc > Struct Template Reference	1778
6.290.1	Detailed Description	1778
6.291	std::cauchy_distribution< _RealType > Class Template Reference	1778
6.291.1	Detailed Description	1779
6.291.2	Member Typedef Documentation	1779
6.291.3	Member Function Documentation	1779
6.291.4	Friends And Related Symbol Documentation	1780
6.292	__gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type > Class Template Reference	1780
6.292.1	Detailed Description	1781
6.292.2	Member Enumeration Documentation	1781
6.292.3	Constructor & Destructor Documentation	1782
6.292.4	Member Function Documentation	1782
6.293	__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc > Class Template Reference	1784
6.293.1	Detailed Description	1785
6.293.2	Constructor & Destructor Documentation	1786
6.294	__gnu_pbds::cc_hash_tag Struct Reference	1789
6.294.1	Detailed Description	1789
6.295	__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > Class Template Reference	1789
6.295.1	Detailed Description	1792
6.295.2	Member Enumeration Documentation	1792
6.295.3	Member Function Documentation	1792
6.296	__gnu_cxx::char_traits< _CharT > Struct Template Reference	1794
6.296.1	Detailed Description	1795
6.297	std::char_traits< _CharT > Struct Template Reference	1795
6.297.1	Detailed Description	1796
6.298	std::char_traits< __gnu_cxx::character< _Value, _Int, _St > > Struct Template Reference	1796
6.298.1	Detailed Description	1797

6.299 std::char_traits< char > Struct Reference	1797
6.299.1 Detailed Description	1797
6.300 std::char_traits< wchar_t > Struct Reference	1798
6.300.1 Detailed Description	1798
6.301 __gnu_cxx::character< _Value, _Int, _St > Struct Template Reference	1798
6.301.1 Detailed Description	1799
6.302 std::chi_squared_distribution< _RealType > Class Template Reference	1799
6.302.1 Detailed Description	1800
6.302.2 Member Typedef Documentation	1800
6.302.3 Member Function Documentation	1800
6.302.4 Friends And Related Symbol Documentation	1801
6.303 std::codecvt< _InternT, _ExternT, _StateT > Class Template Reference	1802
6.303.1 Detailed Description	1803
6.303.2 Member Function Documentation	1804
6.304 std::codecvt< _InternT, _ExternT, encoding_state > Class Template Reference	1807
6.304.1 Detailed Description	1808
6.304.2 Member Function Documentation	1809
6.305 std::codecvt< char, char, mbstate_t > Class Reference	1812
6.305.1 Detailed Description	1813
6.305.2 Member Function Documentation	1814
6.306 std::codecvt< char16_t, char, mbstate_t > Class Reference	1817
6.306.1 Detailed Description	1818
6.306.2 Member Function Documentation	1818
6.307 std::codecvt< char32_t, char, mbstate_t > Class Reference	1822
6.307.1 Detailed Description	1823
6.307.2 Member Function Documentation	1823
6.308 std::codecvt< wchar_t, char, mbstate_t > Class Reference	1827
6.308.1 Detailed Description	1828
6.308.2 Member Function Documentation	1828
6.309 std::codecvt_base Class Reference	1832
6.309.1 Detailed Description	1832
6.310 std::codecvt_byname< _InternT, _ExternT, _StateT > Class Template Reference	1832
6.310.1 Detailed Description	1834
6.310.2 Member Function Documentation	1834
6.311 std::collate< _CharT > Class Template Reference	1838
6.311.1 Detailed Description	1839
6.311.2 Member Typedef Documentation	1839
6.311.3 Constructor & Destructor Documentation	1839
6.311.4 Member Function Documentation	1840

6.311.5 Member Data Documentation	1844
6.312 std::collate_byname< _CharT > Class Template Reference	1844
6.312.1 Detailed Description	1845
6.312.2 Member Typedef Documentation	1845
6.312.3 Member Function Documentation	1846
6.312.4 Member Data Documentation	1848
6.313 std::common_iterator< _It, _Sent > Class Template Reference	1848
6.313.1 Detailed Description	1849
6.314 std::common_type< _Tp > Struct Template Reference	1850
6.314.1 Detailed Description	1850
6.315 std::common_type< chrono::duration< _Rep, _Period > > Struct Template Reference	1850
6.315.1 Detailed Description	1850
6.316 std::common_type< chrono::duration< _Rep, _Period >, chrono::duration< _Rep, _Period > > Struct Template Reference	1850
6.316.1 Detailed Description	1850
6.317 std::common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > > Struct Template Reference	1850
6.317.1 Detailed Description	1851
6.318 std::common_type< chrono::time_point< _Clock, _Duration > > Struct Template Reference	1851
6.318.1 Detailed Description	1851
6.319 std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > > Struct Template Reference	1851
6.319.1 Detailed Description	1851
6.320 std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > > Struct Template Reference	1851
6.320.1 Detailed Description	1852
6.321 std::compare_three_way_result< _Tp, _Up > Struct Template Reference	1852
6.321.1 Detailed Description	1852
6.322 std::complex< _Tp > Class Template Reference	1852
6.322.1 Detailed Description	1853
6.322.2 Member Typedef Documentation	1853
6.322.3 Constructor & Destructor Documentation	1853
6.322.4 Member Function Documentation	1853
6.323 std::complex< double > Class Reference	1854
6.323.1 Detailed Description	1854
6.324 std::complex< float > Class Reference	1854
6.324.1 Detailed Description	1855
6.325 std::complex< long double > Class Reference	1855
6.325.1 Detailed Description	1856
6.326 __gnu_pbds::detail::cond_dealtor< Entry, _Alloc > Class Template Reference	1856

6.326.1 Detailed Description	1857
6.327 <code>__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type ></code> Class Template Reference	1857
6.327.1 Detailed Description	1857
6.328 <code>__gnu_cxx::condition_base</code> Struct Reference	1857
6.328.1 Detailed Description	1858
6.329 <code>std::condition_variable</code> Class Reference	1858
6.329.1 Detailed Description	1858
6.330 <code>std::_V2::condition_variable_any</code> Class Reference	1858
6.330.1 Detailed Description	1859
6.331 <code>std::conditional< _Cond, _Iftrue, _Iffalse ></code> Struct Template Reference	1859
6.331.1 Detailed Description	1859
6.332 <code>__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator</code> Struct Reference	1859
6.332.1 Detailed Description	1860
6.333 <code>std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg ></code> Class Template Reference	1861
6.333.1 Detailed Description	1861
6.333.2 Member Typedef Documentation	1861
6.334 <code>std::const_mem_fun1_t< _Ret, _Tp, _Arg ></code> Class Template Reference	1862
6.334.1 Detailed Description	1862
6.334.2 Member Typedef Documentation	1862
6.335 <code>std::const_mem_fun_ref_t< _Ret, _Tp ></code> Class Template Reference	1863
6.335.1 Detailed Description	1863
6.335.2 Member Typedef Documentation	1863
6.336 <code>std::const_mem_fun_t< _Ret, _Tp ></code> Class Template Reference	1864
6.336.1 Detailed Description	1864
6.336.2 Member Typedef Documentation	1864
6.337 <code>__gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 ></code> Struct Template Reference	1865
6.337.1 Detailed Description	1865
6.338 <code>__gnu_parallel::constant_size_blocks_tag</code> Struct Reference	1865
6.338.1 Detailed Description	1866
6.339 <code>__gnu_cxx::constant_unary_fun< _Result, _Argument ></code> Struct Template Reference	1866
6.339.1 Detailed Description	1866
6.340 <code>__gnu_cxx::constant_void_fun< _Result ></code> Struct Template Reference	1866
6.340.1 Detailed Description	1867
6.341 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI ></code> Struct Template Reference	1867
6.341.1 Detailed Description	1867
6.342 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type ></code> Struct Template Reference	1867
6.342.1 Detailed Description	1867

6.342.2 Member Typedef Documentation	1867
6.343 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type ></code> Struct Template Reference	1867
6.343.1 Detailed Description	1868
6.343.2 Member Typedef Documentation	1868
6.344 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type ></code> Struct Template Reference	1868
6.344.1 Detailed Description	1868
6.344.2 Member Typedef Documentation	1868
6.345 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type ></code> Struct Template Reference	1868
6.345.1 Detailed Description	1869
6.345.2 Member Typedef Documentation	1869
6.346 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type ></code> Struct Template Reference	1869
6.346.1 Detailed Description	1869
6.346.2 Member Typedef Documentation	1869
6.347 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI ></code> Struct Template Reference	1869
6.347.1 Detailed Description	1870
6.347.2 Member Typedef Documentation	1870
6.348 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI ></code> Struct Template Reference	1870
6.348.1 Detailed Description	1870
6.348.2 Member Typedef Documentation	1870
6.349 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI ></code> Struct Template Reference	1870
6.349.1 Detailed Description	1871
6.349.2 Member Typedef Documentation	1871
6.350 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI ></code> Struct Template Reference	1871
6.350.1 Detailed Description	1871
6.350.2 Member Typedef Documentation	1871
6.351 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI ></code> Struct Template Reference	1871
6.351.1 Detailed Description	1872
6.352 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI ></code> Struct Template Reference	1872
6.352.1 Detailed Description	1872
6.352.2 Member Typedef Documentation	1872
6.353 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI ></code> Struct Template Reference	1872

6.353.1 Detailed Description	1872
6.353.2 Member Typedef Documentation	1873
6.354 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI > Struct</code> Template Reference	1873
6.354.1 Detailed Description	1873
6.354.2 Member Typedef Documentation	1873
6.355 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI > Struct</code> Template Reference	1873
6.355.1 Detailed Description	1873
6.355.2 Member Typedef Documentation	1874
6.356 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI ></code> Struct Template Reference	1874
6.356.1 Detailed Description	1874
6.356.2 Member Typedef Documentation	1874
6.357 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI > Struct</code> Template Reference	1874
6.357.1 Detailed Description	1874
6.357.2 Member Typedef Documentation	1875
6.358 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI > Struct</code> Template Reference	1875
6.358.1 Detailed Description	1875
6.358.2 Member Typedef Documentation	1875
6.359 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_TI > Struct</code> Template Reference	1875
6.359.1 Detailed Description	1875
6.360 <code>__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI ></code> Struct Template Reference	1876
6.360.1 Detailed Description	1876
6.360.2 Member Typedef Documentation	1876
6.361 <code>__gnu_pbds::container_error</code> Struct Reference	1876
6.361.1 Detailed Description	1877
6.361.2 Member Function Documentation	1877
6.362 <code>__gnu_pbds::container_tag</code> Struct Reference	1877
6.362.1 Detailed Description	1877
6.363 <code>__gnu_pbds::container_traits< Cntnr > Struct</code> Template Reference	1877
6.363.1 Detailed Description	1878
6.363.2 Member Enumeration Documentation	1878
6.364 <code>__gnu_pbds::container_traits_base< _Tag > Struct</code> Template Reference	1879
6.364.1 Detailed Description	1879
6.365 <code>__gnu_pbds::container_traits_base< binary_heap_tag > Struct</code> Reference	1879
6.365.1 Detailed Description	1879

6.366 __gnu_pbds::container_traits_base< binomial_heap_tag > Struct Reference	1879
6.366.1 Detailed Description	1879
6.367 __gnu_pbds::container_traits_base< cc_hash_tag > Struct Reference	1879
6.367.1 Detailed Description	1880
6.368 __gnu_pbds::container_traits_base< gp_hash_tag > Struct Reference	1880
6.368.1 Detailed Description	1880
6.369 __gnu_pbds::container_traits_base< list_update_tag > Struct Reference	1880
6.369.1 Detailed Description	1880
6.370 __gnu_pbds::container_traits_base< ov_tree_tag > Struct Reference	1880
6.370.1 Detailed Description	1880
6.371 __gnu_pbds::container_traits_base< pairing_heap_tag > Struct Reference	1881
6.371.1 Detailed Description	1881
6.372 __gnu_pbds::container_traits_base< pat_trie_tag > Struct Reference	1881
6.372.1 Detailed Description	1881
6.373 __gnu_pbds::container_traits_base< rb_tree_tag > Struct Reference	1881
6.373.1 Detailed Description	1881
6.374 __gnu_pbds::container_traits_base< rc_binomial_heap_tag > Struct Reference	1881
6.374.1 Detailed Description	1882
6.375 __gnu_pbds::container_traits_base< splay_tree_tag > Struct Reference	1882
6.375.1 Detailed Description	1882
6.376 __gnu_pbds::container_traits_base< thin_heap_tag > Struct Reference	1882
6.376.1 Detailed Description	1882
6.377 std::contiguous_iterator_tag Struct Reference	1882
6.377.1 Detailed Description	1883
6.378 std::counted_iterator< _It > Class Template Reference	1883
6.378.1 Detailed Description	1884
6.379 std::ctype< _CharT > Class Template Reference	1885
6.379.1 Detailed Description	1886
6.379.2 Member Function Documentation	1887
6.379.3 Member Data Documentation	1898
6.380 std::ctype< char > Class Reference	1898
6.380.1 Detailed Description	1900
6.380.2 Member Typedef Documentation	1900
6.380.3 Constructor & Destructor Documentation	1900
6.380.4 Member Function Documentation	1901
6.380.5 Member Data Documentation	1909
6.381 std::ctype< wchar_t > Class Reference	1910
6.381.1 Detailed Description	1912
6.381.2 Member Typedef Documentation	1912

6.381.3 Constructor & Destructor Documentation	1912
6.381.4 Member Function Documentation	1912
6.381.5 Member Data Documentation	1923
6.382 std::ctype_base Struct Reference	1923
6.382.1 Detailed Description	1923
6.383 std::ctype_byname< _CharT > Class Template Reference	1924
6.383.1 Detailed Description	1925
6.383.2 Member Function Documentation	1926
6.383.3 Member Data Documentation	1937
6.384 std::ctype_byname< char > Class Reference	1937
6.384.1 Detailed Description	1939
6.384.2 Member Typedef Documentation	1939
6.384.3 Member Function Documentation	1939
6.384.4 Member Data Documentation	1948
6.385 std::ranges::dangling Struct Reference	1948
6.385.1 Detailed Description	1948
6.386 __gnu_cxx::debug_allocator< _Alloc > Class Template Reference	1948
6.386.1 Detailed Description	1949
6.387 std::decay< _Tp > Class Template Reference	1949
6.387.1 Detailed Description	1949
6.388 std::decimal::decimal128 Class Reference	1949
6.388.1 Detailed Description	1950
6.388.2 Constructor & Destructor Documentation	1951
6.389 std::decimal::decimal32 Class Reference	1951
6.389.1 Detailed Description	1952
6.389.2 Constructor & Destructor Documentation	1952
6.390 std::decimal::decimal64 Class Reference	1952
6.390.1 Detailed Description	1953
6.390.2 Constructor & Destructor Documentation	1954
6.391 simd_abi::deduce< _Tp, _Np,... > Struct Template Reference	1954
6.391.1 Detailed Description	1954
6.392 __gnu_pbds::detail::default_comb_hash_fn Struct Reference	1954
6.392.1 Detailed Description	1954
6.392.2 Member Typedef Documentation	1954
6.393 std::default_delete< _Tp > Struct Template Reference	1955
6.393.1 Detailed Description	1955
6.393.2 Constructor & Destructor Documentation	1955
6.393.3 Member Function Documentation	1955
6.394 std::default_delete< _Tp[]> Struct Template Reference	1955

6.394.1 Detailed Description	1956
6.394.2 Constructor & Destructor Documentation	1956
6.394.3 Member Function Documentation	1956
6.395 <code>__gnu_pbds::detail::default_eq_fn< Key ></code> Struct Template Reference	1956
6.395.1 Detailed Description	1957
6.395.2 Member Typedef Documentation	1957
6.396 <code>__gnu_pbds::detail::default_hash_fn< Key ></code> Struct Template Reference	1957
6.396.1 Detailed Description	1957
6.396.2 Member Typedef Documentation	1957
6.397 <code>__gnu_parallel::default_parallel_tag</code> Struct Reference	1957
6.397.1 Detailed Description	1958
6.397.2 Member Function Documentation	1958
6.398 <code>__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn ></code> Struct Template Reference	1959
6.398.1 Detailed Description	1959
6.398.2 Member Typedef Documentation	1959
6.399 <code>__gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn ></code> Struct Template Reference	1959
6.399.1 Detailed Description	1959
6.399.2 Member Typedef Documentation	1959
6.400 <code>std::default_sentinel_t</code> Struct Reference	1959
6.400.1 Detailed Description	1960
6.401 <code>__gnu_pbds::detail::default_trie_access_traits< Key ></code> Struct Template Reference	1960
6.401.1 Detailed Description	1960
6.402 <code>__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > ></code> Struct Template Reference	1960
6.402.1 Detailed Description	1960
6.402.2 Member Typedef Documentation	1960
6.403 <code>__gnu_pbds::detail::default_update_policy</code> Struct Reference	1961
6.403.1 Detailed Description	1961
6.403.2 Member Typedef Documentation	1961
6.404 <code>std::defer_lock_t</code> Struct Reference	1961
6.404.1 Detailed Description	1961
6.405 <code>std::__debug::deque< _Tp, _Allocator ></code> Class Template Reference	1961
6.405.1 Detailed Description	1963
6.406 <code>std::deque< _Tp, _Alloc ></code> Class Template Reference	1963
6.406.1 Detailed Description	1967
6.406.2 Constructor & Destructor Documentation	1968
6.406.3 Member Function Documentation	1971
6.407 <code>std::destroying_delete_t</code> Struct Reference	1987
6.407.1 Detailed Description	1987

6.408 std::tr2::direct_bases< _Tp > Struct Template Reference	1987
6.408.1 Detailed Description	1987
6.409 __gnu_pbds::direct_mask_range_hashing< Size_Type > Class Template Reference	1987
6.409.1 Detailed Description	1988
6.409.2 Member Function Documentation	1988
6.410 __gnu_pbds::direct_mod_range_hashing< Size_Type > Class Template Reference	1988
6.410.1 Detailed Description	1989
6.410.2 Member Function Documentation	1989
6.411 std::filesystem::directory_entry Class Reference	1989
6.411.1 Detailed Description	1990
6.412 std::filesystem::directory_iterator Class Reference	1990
6.412.1 Detailed Description	1991
6.413 std::discard_block_engine< _RandomNumberEngine, __p, __r > Class Template Reference	1991
6.413.1 Detailed Description	1992
6.413.2 Member Typedef Documentation	1992
6.413.3 Constructor & Destructor Documentation	1993
6.413.4 Member Function Documentation	1994
6.413.5 Friends And Related Symbol Documentation	1995
6.414 std::discrete_distribution< _IntType > Class Template Reference	1996
6.414.1 Detailed Description	1997
6.414.2 Member Typedef Documentation	1997
6.414.3 Member Function Documentation	1997
6.414.4 Friends And Related Symbol Documentation	1998
6.415 std::divides< _Tp > Struct Template Reference	1999
6.415.1 Detailed Description	2000
6.415.2 Member Typedef Documentation	2000
6.416 std::divides< void > Struct Reference	2001
6.416.1 Detailed Description	2001
6.417 std::domain_error Class Reference	2001
6.417.1 Detailed Description	2002
6.417.2 Member Function Documentation	2002
6.418 __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc > Struct Template Reference	2002
6.418.1 Detailed Description	2002
6.419 std::chrono::duration< _Rep, _Period > Class Template Reference	2002
6.419.1 Detailed Description	2003
6.420 std::chrono::duration_values< _Rep > Struct Template Reference	2003
6.420.1 Detailed Description	2004
6.421 std::tr2::dynamic_bitset< _WordT, _Alloc > Class Template Reference	2004
6.421.1 Detailed Description	2006

6.421.2 Constructor & Destructor Documentation	2007
6.421.3 Member Function Documentation	2009
6.422 <code>std::ranges::empty_view< _Tp ></code> Class Template Reference	2018
6.422.1 Detailed Description	2019
6.423 <code>std::enable_if< bool, _Tp ></code> Struct Template Reference	2019
6.423.1 Detailed Description	2019
6.424 <code>std::enable_shared_from_this< _Tp ></code> Class Template Reference	2019
6.424.1 Detailed Description	2020
6.424.2 Member Function Documentation	2020
6.425 <code>__gnu_cxx::enc_filebuf< _CharT ></code> Class Template Reference	2020
6.425.1 Detailed Description	2023
6.425.2 Member Function Documentation	2023
6.425.3 Member Data Documentation	2037
6.426 <code>__gnu_cxx::encoding_char_traits< _CharT ></code> Struct Template Reference	2039
6.426.1 Detailed Description	2041
6.427 <code>__gnu_cxx::encoding_state</code> Class Reference	2041
6.427.1 Detailed Description	2041
6.428 <code>__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, No_Throw ></code> Struct Template Reference	2042
6.428.1 Detailed Description	2042
6.429 <code>__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false ></code> Struct Template Reference	2042
6.429.1 Detailed Description	2042
6.430 <code>__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, true ></code> Struct Template Reference	2042
6.430.1 Detailed Description	2042
6.430.2 Member Typedef Documentation	2042
6.431 <code>__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, No_Throw ></code> Struct Template Reference	2043
6.431.1 Detailed Description	2043
6.432 <code>__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, false ></code> Struct Template Reference	2043
6.432.1 Detailed Description	2043
6.433 <code>__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, true ></code> Struct Template Reference	2043
6.433.1 Detailed Description	2043
6.434 <code>__gnu_pbds::detail::eq_by_less< Key, Cmp_Fn ></code> Struct Template Reference	2044
6.434.1 Detailed Description	2044
6.435 <code>__gnu_parallel::equal_split_tag</code> Struct Reference	2044
6.435.1 Detailed Description	2044
6.436 <code>std::equal_to< _Tp ></code> Struct Template Reference	2044
6.436.1 Detailed Description	2045
6.436.2 Member Typedef Documentation	2045
6.437 <code>std::ranges::equal_to</code> Struct Reference	2046
6.437.1 Detailed Description	2046

6.438 <code>std::equal_to< void ></code> Struct Reference	2046
6.438.1 Detailed Description	2046
6.439 <code>std::_V2::error_category</code> Class Reference	2046
6.439.1 Detailed Description	2047
6.439.2 Member Function Documentation	2047
6.440 <code>std::error_code</code> Class Reference	2048
6.440.1 Detailed Description	2048
6.440.2 Member Function Documentation	2048
6.441 <code>std::error_condition</code> Class Reference	2049
6.441.1 Detailed Description	2050
6.441.2 Constructor & Destructor Documentation	2050
6.441.3 Member Function Documentation	2050
6.442 <code>__gnu_parallel::exact_tag</code> Struct Reference	2051
6.442.1 Detailed Description	2051
6.442.2 Member Function Documentation	2051
6.443 <code>std::exception</code> Class Reference	2052
6.443.1 Detailed Description	2053
6.443.2 Member Function Documentation	2053
6.444 <code>std::__exception_ptr::exception_ptr</code> Class Reference	2053
6.444.1 Detailed Description	2053
6.445 <code>std::exponential_distribution< _RealType ></code> Class Template Reference	2054
6.445.1 Detailed Description	2054
6.445.2 Member Typedef Documentation	2055
6.445.3 Constructor & Destructor Documentation	2055
6.445.4 Member Function Documentation	2055
6.445.5 Friends And Related Symbol Documentation	2056
6.446 <code>std::extent< typename, _UInt ></code> Struct Template Reference	2057
6.446.1 Detailed Description	2057
6.447 <code>std::extreme_value_distribution< _RealType ></code> Class Template Reference	2057
6.447.1 Detailed Description	2058
6.447.2 Member Typedef Documentation	2058
6.447.3 Member Function Documentation	2058
6.447.4 Friends And Related Symbol Documentation	2060
6.448 <code>std::locale::facet</code> Class Reference	2060
6.448.1 Detailed Description	2062
6.448.2 Constructor & Destructor Documentation	2062
6.449 <code>std::ios_base::failure</code> Class Reference	2062
6.449.1 Detailed Description	2063
6.449.2 Member Function Documentation	2063

6.450 std::filesystem::file_status Class Reference	2063
6.450.1 Detailed Description	2064
6.451 std::experimental::filesystem::v1::filesystem_error Class Reference	2064
6.451.1 Detailed Description	2064
6.451.2 Member Function Documentation	2065
6.452 std::filesystem::filesystem_error Class Reference	2065
6.452.1 Detailed Description	2066
6.452.2 Member Function Documentation	2066
6.453 __gnu_parallel::find_tag Struct Reference	2066
6.453.1 Detailed Description	2066
6.454 std::fisher_f_distribution<_RealType> Class Template Reference	2066
6.454.1 Detailed Description	2067
6.454.2 Member Typedef Documentation	2068
6.454.3 Member Function Documentation	2068
6.454.4 Friends And Related Symbol Documentation	2069
6.455 __gnu_cxx::forced_error Struct Reference	2070
6.455.1 Detailed Description	2070
6.455.2 Member Function Documentation	2070
6.456 std::forward_iterator_tag Struct Reference	2070
6.456.1 Detailed Description	2071
6.457 std::__debug::forward_list<_Tp, _Alloc> Class Template Reference	2071
6.457.1 Detailed Description	2073
6.458 std::forward_list<_Tp, _Alloc> Class Template Reference	2074
6.458.1 Detailed Description	2076
6.458.2 Constructor & Destructor Documentation	2076
6.458.3 Member Function Documentation	2079
6.459 std::fpos<_StateT> Class Template Reference	2091
6.459.1 Detailed Description	2091
6.459.2 Constructor & Destructor Documentation	2092
6.459.3 Member Function Documentation	2092
6.460 __gnu_cxx::free_list Class Reference	2093
6.460.1 Detailed Description	2094
6.460.2 Member Function Documentation	2094
6.461 std::from_chars_result Struct Reference	2094
6.461.1 Detailed Description	2094
6.462 std::front_insert_iterator<_Container> Class Template Reference	2095
6.462.1 Detailed Description	2095
6.462.2 Member Typedef Documentation	2096
6.462.3 Constructor & Destructor Documentation	2096

6.462.4 Member Function Documentation	2096
6.463 <code>std::function< _Res(_ArgTypes...)></code> Class Template Reference	2097
6.463.1 Detailed Description	2098
6.463.2 Constructor & Destructor Documentation	2098
6.463.3 Member Function Documentation	2100
6.464 <code>std::future< _Res ></code> Class Template Reference	2103
6.464.1 Detailed Description	2104
6.464.2 Constructor & Destructor Documentation	2104
6.464.3 Member Function Documentation	2104
6.465 <code>std::future< _Res & ></code> Class Template Reference	2105
6.465.1 Detailed Description	2105
6.465.2 Constructor & Destructor Documentation	2106
6.465.3 Member Function Documentation	2106
6.466 <code>std::future< void ></code> Class Reference	2106
6.466.1 Detailed Description	2107
6.466.2 Constructor & Destructor Documentation	2107
6.466.3 Member Function Documentation	2107
6.467 <code>std::future_error</code> Class Reference	2108
6.467.1 Detailed Description	2108
6.467.2 Member Function Documentation	2108
6.468 <code>std::gamma_distribution< _RealType ></code> Class Template Reference	2108
6.468.1 Detailed Description	2109
6.468.2 Member Typedef Documentation	2110
6.468.3 Constructor & Destructor Documentation	2110
6.468.4 Member Function Documentation	2110
6.468.5 Friends And Related Symbol Documentation	2111
6.469 <code>std::geometric_distribution< _IntType ></code> Class Template Reference	2112
6.469.1 Detailed Description	2113
6.469.2 Member Typedef Documentation	2113
6.469.3 Member Function Documentation	2113
6.469.4 Friends And Related Symbol Documentation	2114
6.470 <code>__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc ></code> Class Template Reference	2115
6.470.1 Detailed Description	2116
6.470.2 Constructor & Destructor Documentation	2116
6.471 <code>__gnu_pbds::gp_hash_tag</code> Struct Reference	2120
6.471.1 Detailed Description	2121
6.472 <code>__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy ></code> Class Template Reference	2121
6.472.1 Detailed Description	2123

6.472.2 Member Enumeration Documentation	2123
6.472.3 Member Function Documentation	2123
6.473 std::greater< _Tp > Struct Template Reference	2125
6.473.1 Detailed Description	2126
6.473.2 Member Typedef Documentation	2126
6.474 std::ranges::greater Struct Reference	2126
6.474.1 Detailed Description	2126
6.475 std::greater< void > Struct Reference	2126
6.475.1 Detailed Description	2127
6.476 std::greater_equal< _Tp > Struct Template Reference	2127
6.476.1 Detailed Description	2127
6.476.2 Member Typedef Documentation	2128
6.477 std::ranges::greater_equal Struct Reference	2128
6.477.1 Detailed Description	2128
6.478 std::greater_equal< void > Struct Reference	2128
6.478.1 Detailed Description	2129
6.479 __gnu_cxx::random_condition::group_adjustor Struct Reference	2129
6.479.1 Detailed Description	2129
6.480 __gnu_parallel::growing_blocks_tag Struct Reference	2129
6.480.1 Detailed Description	2129
6.481 std::gslice Class Reference	2129
6.481.1 Detailed Description	2130
6.482 std::gslice_array< _Tp > Class Template Reference	2130
6.482.1 Detailed Description	2131
6.482.2 Member Function Documentation	2131
6.483 std::has_unique_object_representations< _Tp > Struct Template Reference	2133
6.483.1 Detailed Description	2133
6.484 std::has_virtual_destructor< _Tp > Struct Template Reference	2134
6.484.1 Detailed Description	2134
6.485 std::hash< _Tp > Struct Template Reference	2134
6.485.1 Detailed Description	2134
6.486 std::hash< __debug::bitset< _Nb > > Struct Template Reference	2135
6.486.1 Detailed Description	2135
6.487 std::hash< __debug::vector< bool, _Alloc > > Struct Template Reference	2135
6.487.1 Detailed Description	2135
6.488 std::hash< __gnu_cxx::__u16vstring > Struct Reference	2135
6.488.1 Detailed Description	2136
6.489 std::hash< __gnu_cxx::__u32vstring > Struct Reference	2136
6.489.1 Detailed Description	2136

6.490 std::hash< __gnu_cxx::__vstring > Struct Reference	2136
6.490.1 Detailed Description	2136
6.491 std::hash< __gnu_cxx::__wvstring > Struct Reference	2137
6.491.1 Detailed Description	2137
6.492 std::hash< __gnu_cxx::throw_value_limit > Struct Reference	2137
6.492.1 Detailed Description	2137
6.492.2 Member Typedef Documentation	2138
6.493 std::hash< __gnu_cxx::throw_value_random > Struct Reference	2138
6.493.1 Detailed Description	2138
6.493.2 Member Typedef Documentation	2138
6.494 std::hash< __gnu_debug::basic_string< _CharT > > Struct Template Reference	2139
6.494.1 Detailed Description	2139
6.495 std::hash< __shared_ptr< _Tp, _Lp > > Struct Template Reference	2139
6.495.1 Detailed Description	2140
6.496 std::hash< _Tp * > Struct Template Reference	2140
6.496.1 Detailed Description	2140
6.497 std::hash< bool > Struct Reference	2140
6.497.1 Detailed Description	2140
6.498 std::hash< char > Struct Reference	2141
6.498.1 Detailed Description	2141
6.499 std::hash< char16_t > Struct Reference	2141
6.499.1 Detailed Description	2141
6.500 std::hash< char32_t > Struct Reference	2141
6.500.1 Detailed Description	2142
6.501 std::hash< double > Struct Reference	2142
6.501.1 Detailed Description	2142
6.502 std::hash< error_code > Struct Reference	2142
6.502.1 Detailed Description	2142
6.503 std::hash< error_condition > Struct Reference	2142
6.503.1 Detailed Description	2143
6.504 std::hash< experimental::optional< _Tp > > Struct Template Reference	2143
6.504.1 Detailed Description	2143
6.505 std::hash< experimental::shared_ptr< _Tp > > Struct Template Reference	2143
6.505.1 Detailed Description	2143
6.506 std::hash< float > Struct Reference	2144
6.506.1 Detailed Description	2144
6.507 std::hash< int > Struct Reference	2144
6.507.1 Detailed Description	2144
6.508 std::hash< long > Struct Reference	2144

6.508.1 Detailed Description	2145
6.509 std::hash< long double > Struct Reference	2145
6.509.1 Detailed Description	2145
6.510 std::hash< long long > Struct Reference	2145
6.510.1 Detailed Description	2145
6.511 std::hash< shared_ptr< _Tp > > Struct Template Reference	2145
6.511.1 Detailed Description	2146
6.512 std::hash< short > Struct Reference	2146
6.512.1 Detailed Description	2146
6.513 std::hash< signed char > Struct Reference	2146
6.513.1 Detailed Description	2147
6.514 std::hash< string > Struct Reference	2147
6.514.1 Detailed Description	2147
6.515 std::hash< thread::id > Struct Reference	2147
6.515.1 Detailed Description	2147
6.516 std::hash< type_index > Struct Reference	2147
6.516.1 Detailed Description	2148
6.517 std::hash< u16string > Struct Reference	2148
6.517.1 Detailed Description	2148
6.518 std::hash< u32string > Struct Reference	2148
6.518.1 Detailed Description	2148
6.519 std::hash< unique_ptr< _Tp, _Dp > > Struct Template Reference	2148
6.519.1 Detailed Description	2149
6.520 std::hash< unsigned char > Struct Reference	2149
6.520.1 Detailed Description	2149
6.521 std::hash< unsigned int > Struct Reference	2149
6.521.1 Detailed Description	2149
6.522 std::hash< unsigned long > Struct Reference	2150
6.522.1 Detailed Description	2150
6.523 std::hash< unsigned long long > Struct Reference	2150
6.523.1 Detailed Description	2150
6.524 std::hash< unsigned short > Struct Reference	2150
6.524.1 Detailed Description	2151
6.525 std::hash< wchar_t > Struct Reference	2151
6.525.1 Detailed Description	2151
6.526 std::hash< wstring > Struct Reference	2151
6.526.1 Detailed Description	2151
6.527 std::hash<::bitset< _Nb > > Struct Template Reference	2151
6.527.1 Detailed Description	2152

6.528 <code>std::hash<::vector< bool, _Alloc > ></code> Struct Template Reference	2152
6.528.1 Detailed Description	2152
6.529 <code>__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash ></code> Struct Template Reference	2153
6.529.1 Detailed Description	2153
6.530 <code>__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false ></code> Struct Template Reference	2153
6.530.1 Detailed Description	2153
6.531 <code>__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true ></code> Struct Template Reference	2154
6.531.1 Detailed Description	2154
6.532 <code>__gnu_pbds::hash_exponential_size_policy< Size_Type ></code> Class Template Reference	2154
6.532.1 Detailed Description	2155
6.532.2 Constructor & Destructor Documentation	2155
6.533 <code>__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type ></code> Class Template Reference	2155
6.533.1 Detailed Description	2156
6.533.2 Member Enumeration Documentation	2156
6.533.3 Constructor & Destructor Documentation	2156
6.533.4 Member Function Documentation	2157
6.534 <code>__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size ></code> Class Template Reference	2157
6.534.1 Detailed Description	2158
6.535 <code>__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true ></code> Class Template Reference	2158
6.535.1 Detailed Description	2158
6.536 <code>__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc ></code> Class Template Reference	2158
6.536.1 Detailed Description	2160
6.537 <code>__gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc ></code> Class Template Reference	2160
6.537.1 Detailed Description	2161
6.538 <code>__gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc ></code> Class Template Reference	2161
6.538.1 Detailed Description	2163
6.539 <code>__gnu_pbds::hash_prime_size_policy</code> Class Reference	2163
6.539.1 Detailed Description	2163
6.539.2 Member Typedef Documentation	2163
6.539.3 Constructor & Destructor Documentation	2163
6.540 <code>__gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc ></code> Class Template Reference	2164
6.540.1 Detailed Description	2165
6.541 <code>__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type ></code> Class Template Reference	2165
6.541.1 Detailed Description	2166
6.541.2 Member Enumeration Documentation	2166
6.541.3 Constructor & Destructor Documentation	2167

6.541.4 Member Function Documentation	2167
6.542 std::locale::id Class Reference	2169
6.542.1 Detailed Description	2169
6.542.2 Constructor & Destructor Documentation	2169
6.542.3 Friends And Related Symbol Documentation	2169
6.543 std::thread::id Class Reference	2170
6.543.1 Detailed Description	2171
6.544 std::identity Struct Reference	2171
6.544.1 Detailed Description	2171
6.545 std::experimental::fundamentals_v1::in_place_t Struct Reference	2171
6.545.1 Detailed Description	2171
6.546 std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType > Class Template Reference	2172
6.546.1 Detailed Description	2172
6.546.2 Member Typedef Documentation	2172
6.546.3 Constructor & Destructor Documentation	2173
6.546.4 Member Function Documentation	2174
6.546.5 Friends And Related Symbol Documentation	2175
6.547 std::indirect_array<_Tp > Class Template Reference	2176
6.547.1 Detailed Description	2177
6.547.2 Member Function Documentation	2177
6.548 std::initializer_list<_E > Class Template Reference	2178
6.548.1 Detailed Description	2179
6.548.2 Friends And Related Symbol Documentation	2179
6.549 std::input_iterator_tag Struct Reference	2180
6.549.1 Detailed Description	2180
6.550 __gnu_pbds::insert_error Struct Reference	2180
6.550.1 Detailed Description	2181
6.550.2 Member Function Documentation	2181
6.551 std::insert_iterator<_Container > Class Template Reference	2181
6.551.1 Detailed Description	2182
6.551.2 Member Typedef Documentation	2183
6.551.3 Constructor & Destructor Documentation	2183
6.551.4 Member Function Documentation	2183
6.552 std::integer_sequence<_Tp, _Idx > Struct Template Reference	2184
6.552.1 Detailed Description	2184
6.553 std::integral_constant<_Tp, __v > Struct Template Reference	2185
6.553.1 Detailed Description	2186
6.554 std::invalid_argument Class Reference	2186
6.554.1 Detailed Description	2187

6.554.2 Member Function Documentation	2187
6.555 std::invoke_result< _Functor, _ArgTypes > Struct Template Reference	2187
6.555.1 Detailed Description	2187
6.556 std::ios_base Class Reference	2187
6.556.1 Detailed Description	2189
6.556.2 Member Typedef Documentation	2190
6.556.3 Member Enumeration Documentation	2191
6.556.4 Constructor & Destructor Documentation	2192
6.556.5 Member Function Documentation	2192
6.556.6 Member Data Documentation	2197
6.557 std::is_abstract< _Tp > Struct Template Reference	2202
6.557.1 Detailed Description	2202
6.558 std::is_aggregate< _Tp > Struct Template Reference	2203
6.558.1 Detailed Description	2203
6.559 std::is_arithmetic< _Tp > Struct Template Reference	2203
6.559.1 Detailed Description	2204
6.560 std::is_array< typename > Struct Template Reference	2204
6.560.1 Detailed Description	2204
6.561 std::is_assignable< _Tp, _Up > Struct Template Reference	2204
6.561.1 Detailed Description	2205
6.562 std::is_base_of< _Base, _Derived > Struct Template Reference	2205
6.562.1 Detailed Description	2205
6.563 std::is_bind_expression< _Tp > Struct Template Reference	2206
6.563.1 Detailed Description	2206
6.564 std::is_bind_expression< _Bind< _Signature > > Struct Template Reference	2207
6.564.1 Detailed Description	2207
6.565 std::is_bind_expression< _Bind_result< _Result, _Signature > > Struct Template Reference	2208
6.565.1 Detailed Description	2208
6.566 std::is_bind_expression< const _Bind< _Signature > > Struct Template Reference	2209
6.566.1 Detailed Description	2209
6.567 std::is_bind_expression< const _Bind_result< _Result, _Signature > > Struct Template Reference	2210
6.567.1 Detailed Description	2210
6.568 std::is_bind_expression< const volatile _Bind< _Signature > > Struct Template Reference	2211
6.568.1 Detailed Description	2211
6.569 std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > > Struct Template Reference	2212
6.569.1 Detailed Description	2212
6.570 std::is_bind_expression< volatile _Bind< _Signature > > Struct Template Reference	2213
6.570.1 Detailed Description	2213

6.571 std::is_bind_expression< volatile _Bind_result< _Result, _Signature > > Struct Template Reference	2214
6.571.1 Detailed Description	2214
6.572 std::is_bounded_array< _Tp > Struct Template Reference	2214
6.572.1 Detailed Description	2214
6.573 std::is_class< _Tp > Struct Template Reference	2215
6.573.1 Detailed Description	2215
6.574 std::is_compound< _Tp > Struct Template Reference	2215
6.574.1 Detailed Description	2216
6.575 std::is_const< typename > Struct Template Reference	2216
6.575.1 Detailed Description	2216
6.576 std::is_constructible< _Tp, _Args > Struct Template Reference	2216
6.576.1 Detailed Description	2217
6.577 std::is_convertible< _From, _To > Struct Template Reference	2217
6.577.1 Detailed Description	2217
6.578 std::is_copy_assignable< _Tp > Struct Template Reference	2217
6.578.1 Detailed Description	2217
6.579 std::is_copy_constructible< _Tp > Struct Template Reference	2217
6.579.1 Detailed Description	2217
6.580 std::is_default_constructible< _Tp > Struct Template Reference	2217
6.580.1 Detailed Description	2218
6.581 std::is_destructible< _Tp > Struct Template Reference	2218
6.581.1 Detailed Description	2218
6.582 std::is_empty< _Tp > Struct Template Reference	2218
6.582.1 Detailed Description	2219
6.583 std::is_enum< _Tp > Struct Template Reference	2219
6.583.1 Detailed Description	2219
6.584 std::is_error_code_enum< _Tp > Struct Template Reference	2220
6.584.1 Detailed Description	2220
6.585 std::is_error_code_enum< future_errc > Struct Reference	2221
6.585.1 Detailed Description	2221
6.586 std::is_error_condition_enum< _Tp > Struct Template Reference	2222
6.586.1 Detailed Description	2222
6.587 std::is_final< _Tp > Struct Template Reference	2223
6.587.1 Detailed Description	2223
6.588 std::is_floating_point< _Tp > Struct Template Reference	2223
6.588.1 Detailed Description	2223
6.589 std::is_function< _Tp > Struct Template Reference	2224
6.589.1 Detailed Description	2224
6.590 std::is_fundamental< _Tp > Struct Template Reference	2224

6.590.1 Detailed Description	2224
6.591 std::is_integral< _Tp > Struct Template Reference	2224
6.591.1 Detailed Description	2224
6.592 std::is_invocable< _Fn, _ArgTypes > Struct Template Reference	2224
6.592.1 Detailed Description	2224
6.593 std::is_invocable_r< _Ret, _Fn, _ArgTypes > Struct Template Reference	2225
6.593.1 Detailed Description	2225
6.594 std::is_layout_compatible< _Tp, _Up > Struct Template Reference	2225
6.594.1 Detailed Description	2225
6.595 std::is_literal_type< _Tp > Struct Template Reference	2226
6.595.1 Detailed Description	2226
6.596 std::is_lvalue_reference< typename > Struct Template Reference	2227
6.596.1 Detailed Description	2227
6.597 std::is_member_function_pointer< _Tp > Struct Template Reference	2228
6.597.1 Detailed Description	2228
6.598 std::is_member_object_pointer< _Tp > Struct Template Reference	2229
6.598.1 Detailed Description	2229
6.599 std::is_member_pointer< _Tp > Struct Template Reference	2229
6.599.1 Detailed Description	2229
6.600 std::is_move_assignable< _Tp > Struct Template Reference	2230
6.600.1 Detailed Description	2230
6.601 std::is_move_constructible< _Tp > Struct Template Reference	2230
6.601.1 Detailed Description	2230
6.602 std::is_nothrow_assignable< _Tp, _Up > Struct Template Reference	2230
6.602.1 Detailed Description	2230
6.603 std::is_nothrow_constructible< _Tp, _Args > Struct Template Reference	2230
6.603.1 Detailed Description	2230
6.604 std::is_nothrow_convertible< _From, _To > Struct Template Reference	2231
6.604.1 Detailed Description	2231
6.605 std::is_nothrow_copy_assignable< _Tp > Struct Template Reference	2231
6.605.1 Detailed Description	2231
6.606 std::is_nothrow_copy_constructible< _Tp > Struct Template Reference	2231
6.606.1 Detailed Description	2231
6.607 std::is_nothrow_default_constructible< _Tp > Struct Template Reference	2231
6.607.1 Detailed Description	2231
6.608 std::is_nothrow_destructible< _Tp > Struct Template Reference	2231
6.608.1 Detailed Description	2232
6.609 std::is_nothrow_invocable< _Fn, _ArgTypes > Struct Template Reference	2232
6.609.1 Detailed Description	2232

6.610 std::is_nothrow_invocable_r<_Ret, _Fn, _ArgTypes > Struct Template Reference	2232
6.610.1 Detailed Description	2232
6.611 std::is_nothrow_move_assignable<_Tp > Struct Template Reference	2232
6.611.1 Detailed Description	2232
6.612 std::is_nothrow_move_constructible<_Tp > Struct Template Reference	2232
6.612.1 Detailed Description	2233
6.613 std::is_nothrow_swappable<_Tp > Struct Template Reference	2233
6.613.1 Detailed Description	2233
6.614 std::is_nothrow_swappable_with<_Tp, _Up > Struct Template Reference	2233
6.614.1 Detailed Description	2233
6.615 std::is_null_pointer<_Tp > Struct Template Reference	2234
6.615.1 Detailed Description	2234
6.616 std::is_object<_Tp > Struct Template Reference	2234
6.616.1 Detailed Description	2235
6.617 std::is_placeholder<_Tp > Struct Template Reference	2235
6.617.1 Detailed Description	2235
6.618 std::is_placeholder<_Placeholder<_Num > > Struct Template Reference	2236
6.618.1 Detailed Description	2236
6.619 std::is_pod<_Tp > Struct Template Reference	2237
6.619.1 Detailed Description	2237
6.620 std::is_pointer<_Tp > Struct Template Reference	2238
6.620.1 Detailed Description	2238
6.621 std::is_pointer_interconvertible_base_of<_Base, _Derived > Struct Template Reference	2239
6.621.1 Detailed Description	2239
6.622 std::is_polymorphic<_Tp > Struct Template Reference	2240
6.622.1 Detailed Description	2240
6.623 std::is_reference<_Tp > Struct Template Reference	2240
6.623.1 Detailed Description	2240
6.624 std::is_rvalue_reference<typename > Struct Template Reference	2241
6.624.1 Detailed Description	2241
6.625 std::is_same<_Tp, _Up > Struct Template Reference	2242
6.625.1 Detailed Description	2242
6.626 std::is_scalar<_Tp > Struct Template Reference	2242
6.626.1 Detailed Description	2242
6.627 std::is_signed<_Tp > Struct Template Reference	2243
6.627.1 Detailed Description	2243
6.628 std::is_standard_layout<_Tp > Struct Template Reference	2243
6.628.1 Detailed Description	2243
6.629 std::is_swappable<_Tp > Struct Template Reference	2244

6.629.1 Detailed Description	2244
6.630 std::is_swappable_with< _Tp, _Up > Struct Template Reference	2244
6.630.1 Detailed Description	2244
6.631 std::is_trivial< _Tp > Struct Template Reference	2244
6.631.1 Detailed Description	2245
6.632 std::is_trivially_assignable< _Tp, _Up > Struct Template Reference	2245
6.632.1 Detailed Description	2245
6.633 std::is_trivially_constructible< _Tp, _Args > Struct Template Reference	2245
6.633.1 Detailed Description	2245
6.634 std::is_trivially_copy_assignable< _Tp > Struct Template Reference	2245
6.634.1 Detailed Description	2245
6.635 std::is_trivially_copy_constructible< _Tp > Struct Template Reference	2245
6.635.1 Detailed Description	2246
6.636 std::is_trivially_copyable< _Tp > Struct Template Reference	2246
6.636.1 Detailed Description	2246
6.637 std::is_trivially_default_constructible< _Tp > Struct Template Reference	2247
6.637.1 Detailed Description	2247
6.638 std::is_trivially_destructible< _Tp > Struct Template Reference	2247
6.638.1 Detailed Description	2247
6.639 std::is_trivially_move_assignable< _Tp > Struct Template Reference	2247
6.639.1 Detailed Description	2247
6.640 std::is_trivially_move_constructible< _Tp > Struct Template Reference	2247
6.640.1 Detailed Description	2247
6.641 std::is_unbounded_array< _Tp > Struct Template Reference	2247
6.641.1 Detailed Description	2248
6.642 std::is_union< _Tp > Struct Template Reference	2248
6.642.1 Detailed Description	2248
6.643 std::is_unsigned< _Tp > Struct Template Reference	2249
6.643.1 Detailed Description	2249
6.644 std::is_void< _Tp > Struct Template Reference	2249
6.644.1 Detailed Description	2249
6.645 std::is_volatile< typename > Struct Template Reference	2249
6.645.1 Detailed Description	2250
6.646 std::istream_iterator< _Tp, _CharT, _Traits, _Dist > Class Template Reference	2250
6.646.1 Detailed Description	2251
6.646.2 Member Typedef Documentation	2251
6.646.3 Constructor & Destructor Documentation	2252
6.646.4 Friends And Related Symbol Documentation	2252
6.647 std::istreambuf_iterator< _CharT, _Traits > Class Template Reference	2252

6.647.1 Detailed Description	2253
6.647.2 Member Typedef Documentation	2254
6.647.3 Constructor & Destructor Documentation	2255
6.647.4 Member Function Documentation	2255
6.648 __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>::iterator Struct Reference	2256
6.648.1 Detailed Description	2257
6.649 std::experimental::filesystem::v1::path::iterator Class Reference	2257
6.649.1 Detailed Description	2257
6.650 std::filesystem::path::iterator Class Reference	2257
6.650.1 Detailed Description	2258
6.651 std::iterator<_Category, _Tp, _Distance, _Pointer, _Reference> Struct Template Reference	2258
6.651.1 Detailed Description	2259
6.651.2 Member Typedef Documentation	2259
6.652 std::iterator_traits<_Iterator> Struct Template Reference	2260
6.652.1 Detailed Description	2260
6.653 std::iterator_traits<_Tp*> Struct Template Reference	2260
6.653.1 Detailed Description	2261
6.654 __gnu_pbds::join_error Struct Reference	2261
6.654.1 Detailed Description	2261
6.654.2 Member Function Documentation	2261
6.655 std::jthread Class Reference	2262
6.655.1 Detailed Description	2262
6.656 __gnu_pbds::detail::left_child_next_sibling_heap<Value_Type, Cmp_Fn, Node_Metadata, _Alloc> Class Template Reference	2263
6.656.1 Detailed Description	2264
6.657 __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator<Node, _Alloc> Class Template Reference	2264
6.657.1 Detailed Description	2265
6.657.2 Member Typedef Documentation	2265
6.657.3 Constructor & Destructor Documentation	2266
6.657.4 Member Function Documentation	2266
6.658 __gnu_pbds::detail::left_child_next_sibling_heap_node<_Value, _Metadata, _Alloc> Struct Template Reference	2267
6.658.1 Detailed Description	2268
6.659 __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator<Node, _Alloc> Class Template Reference	2268
6.659.1 Detailed Description	2269
6.659.2 Member Typedef Documentation	2269
6.659.3 Constructor & Destructor Documentation	2270
6.659.4 Member Function Documentation	2270

6.660 <code>std::length_error</code> Class Reference	2271
6.660.1 Detailed Description	2271
6.660.2 Member Function Documentation	2272
6.661 <code>std::less< _Tp ></code> Struct Template Reference	2272
6.661.1 Detailed Description	2272
6.661.2 Member Typedef Documentation	2272
6.662 <code>std::ranges::less</code> Struct Reference	2273
6.662.1 Detailed Description	2273
6.663 <code>std::less< void ></code> Struct Reference	2273
6.663.1 Detailed Description	2273
6.664 <code>std::less_equal< _Tp ></code> Struct Template Reference	2274
6.664.1 Detailed Description	2274
6.664.2 Member Typedef Documentation	2274
6.665 <code>std::ranges::less_equal</code> Struct Reference	2275
6.665.1 Detailed Description	2275
6.666 <code>std::less_equal< void ></code> Struct Reference	2275
6.666.1 Detailed Description	2275
6.667 <code>__gnu_cxx::limit_condition::limit_adjustor</code> Struct Reference	2275
6.667.1 Detailed Description	2276
6.668 <code>__gnu_cxx::limit_condition</code> Struct Reference	2276
6.668.1 Detailed Description	2276
6.669 <code>std::linear_congruential_engine< _UIntType, __a, __c, __m ></code> Class Template Reference	2276
6.669.1 Detailed Description	2277
6.669.2 Member Typedef Documentation	2277
6.669.3 Constructor & Destructor Documentation	2278
6.669.4 Member Function Documentation	2278
6.669.5 Friends And Related Symbol Documentation	2280
6.669.6 Member Data Documentation	2281
6.670 <code>__gnu_pbds::linear_probe_fn< Size_Type ></code> Class Template Reference	2281
6.670.1 Detailed Description	2282
6.670.2 Member Function Documentation	2282
6.671 <code>std::__debug::list< _Tp, _Allocator ></code> Class Template Reference	2282
6.671.1 Detailed Description	2284
6.672 <code>std::list< _Tp, _Alloc ></code> Class Template Reference	2284
6.672.1 Detailed Description	2288
6.672.2 Constructor & Destructor Documentation	2288
6.672.3 Member Function Documentation	2291
6.673 <code>__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc ></code> Class Template Reference	2306
6.673.1 Detailed Description	2307

6.673.2 Constructor & Destructor Documentation	2307
6.674 __gnu_pbds::list_update_tag Struct Reference	2307
6.674.1 Detailed Description	2308
6.675 std::locale Class Reference	2308
6.675.1 Detailed Description	2309
6.675.2 Member Typedef Documentation	2309
6.675.3 Constructor & Destructor Documentation	2310
6.675.4 Member Function Documentation	2312
6.675.5 Friends And Related Symbol Documentation	2314
6.675.6 Member Data Documentation	2315
6.676 std::lock_guard< _Mutex > Class Template Reference	2317
6.676.1 Detailed Description	2317
6.677 std::logic_error Class Reference	2317
6.677.1 Detailed Description	2318
6.677.2 Constructor & Destructor Documentation	2318
6.677.3 Member Function Documentation	2318
6.678 std::logical_and< _Tp > Struct Template Reference	2318
6.678.1 Detailed Description	2319
6.678.2 Member Typedef Documentation	2319
6.679 std::logical_and< void > Struct Reference	2320
6.679.1 Detailed Description	2320
6.680 std::logical_not< _Tp > Struct Template Reference	2320
6.680.1 Detailed Description	2320
6.680.2 Member Typedef Documentation	2321
6.681 std::logical_not< void > Struct Reference	2321
6.681.1 Detailed Description	2321
6.682 std::logical_or< _Tp > Struct Template Reference	2321
6.682.1 Detailed Description	2322
6.682.2 Member Typedef Documentation	2322
6.683 std::logical_or< void > Struct Reference	2322
6.683.1 Detailed Description	2322
6.684 std::lognormal_distribution< _RealType > Class Template Reference	2323
6.684.1 Detailed Description	2323
6.684.2 Member Typedef Documentation	2324
6.684.3 Member Function Documentation	2324
6.684.4 Friends And Related Symbol Documentation	2325
6.685 __gnu_pbds::detail::lu_counter_metadata< Size_Type > Class Template Reference	2326
6.685.1 Detailed Description	2326
6.686 __gnu_pbds::lu_counter_policy< Max_Count, _Alloc > Class Template Reference	2326

6.686.1 Detailed Description	2327
6.686.2 Member Typedef Documentation	2327
6.686.3 Member Enumeration Documentation	2327
6.686.4 Member Function Documentation	2327
6.687 <code>__gnu_pbds::detail::lu_counter_policy_base< Size_Type ></code> Class Template Reference	2328
6.687.1 Detailed Description	2328
6.688 <code>__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy ></code> Class Template Reference	2328
6.688.1 Detailed Description	2330
6.689 <code>__gnu_pbds::lu_move_to_front_policy< _Alloc ></code> Class Template Reference	2331
6.689.1 Detailed Description	2331
6.689.2 Member Typedef Documentation	2331
6.689.3 Member Function Documentation	2331
6.690 <code>std::make_signed< _Tp ></code> Struct Template Reference	2332
6.690.1 Detailed Description	2332
6.691 <code>std::make_unsigned< _Tp ></code> Struct Template Reference	2332
6.691.1 Detailed Description	2332
6.692 <code>__gnu_cxx::malloc_allocator< _Tp ></code> Class Template Reference	2332
6.692.1 Detailed Description	2333
6.693 <code>std::__debug::map< _Key, _Tp, _Compare, _Allocator ></code> Class Template Reference	2333
6.693.1 Detailed Description	2336
6.694 <code>std::map< _Key, _Tp, _Compare, _Alloc ></code> Class Template Reference	2336
6.694.1 Detailed Description	2339
6.694.2 Constructor & Destructor Documentation	2340
6.694.3 Member Function Documentation	2343
6.695 <code>std::mask_array< _Tp ></code> Class Template Reference	2364
6.695.1 Detailed Description	2365
6.695.2 Member Function Documentation	2365
6.696 <code>__gnu_pbds::detail::mask_based_range_hashing< Size_Type ></code> Class Template Reference	2367
6.696.1 Detailed Description	2367
6.697 <code>std::match_results< _Bi_iter, _Alloc ></code> Class Template Reference	2368
6.697.1 Detailed Description	2370
6.697.2 Constructor & Destructor Documentation	2370
6.697.3 Member Function Documentation	2371
6.698 <code>__gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash ></code> Struct Template Reference	2376
6.698.1 Detailed Description	2376
6.699 <code>__gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash ></code> Struct Template Reference	2376
6.699.1 Detailed Description	2377
6.700 <code>std::mem_fun1_ref_t< _Ret, _Tp, _Arg ></code> Class Template Reference	2377
6.700.1 Detailed Description	2377

6.700.2 Member Typedef Documentation	2377
6.701 std::mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference	2378
6.701.1 Detailed Description	2378
6.701.2 Member Typedef Documentation	2379
6.702 std::mem_fun_ref_t< _Ret, _Tp > Class Template Reference	2379
6.702.1 Detailed Description	2379
6.702.2 Member Typedef Documentation	2380
6.703 std::mem_fun_t< _Ret, _Tp > Class Template Reference	2380
6.703.1 Detailed Description	2380
6.703.2 Member Typedef Documentation	2381
6.704 std::pmr::memory_resource Class Reference	2381
6.704.1 Detailed Description	2381
6.705 std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > Class Template Reference	2381
6.705.1 Detailed Description	2383
6.705.2 Member Typedef Documentation	2383
6.705.3 Constructor & Destructor Documentation	2384
6.705.4 Member Function Documentation	2384
6.705.5 Friends And Related Symbol Documentation	2384
6.706 std::messages< _CharT > Class Template Reference	2386
6.706.1 Detailed Description	2387
6.706.2 Member Typedef Documentation	2388
6.706.3 Constructor & Destructor Documentation	2388
6.706.4 Member Function Documentation	2389
6.706.5 Member Data Documentation	2389
6.707 std::messages_base Struct Reference	2389
6.707.1 Detailed Description	2389
6.708 std::messages_byname< _CharT > Class Template Reference	2390
6.708.1 Detailed Description	2391
6.708.2 Member Function Documentation	2391
6.708.3 Member Data Documentation	2391
6.709 std::minus< _Tp > Struct Template Reference	2391
6.709.1 Detailed Description	2392
6.709.2 Member Typedef Documentation	2392
6.710 std::minus< void > Struct Reference	2393
6.710.1 Detailed Description	2393
6.711 __gnu_pbds::detail::mod_based_range_hashing< Size_Type > Class Template Reference	2393
6.711.1 Detailed Description	2394
6.712 std::modulus< _Tp > Struct Template Reference	2394

6.712.1 Detailed Description	2394
6.712.2 Member Typedef Documentation	2394
6.713 std::modulus< void > Struct Reference	2395
6.713.1 Detailed Description	2395
6.714 std::money_base Class Reference	2395
6.714.1 Detailed Description	2396
6.715 std::money_get< _CharT, _InIter > Class Template Reference	2397
6.715.1 Detailed Description	2398
6.715.2 Member Typedef Documentation	2398
6.715.3 Constructor & Destructor Documentation	2398
6.715.4 Member Function Documentation	2399
6.715.5 Member Data Documentation	2401
6.716 std::money_put< _CharT, _OutIter > Class Template Reference	2401
6.716.1 Detailed Description	2402
6.716.2 Member Typedef Documentation	2402
6.716.3 Constructor & Destructor Documentation	2402
6.716.4 Member Function Documentation	2403
6.716.5 Member Data Documentation	2405
6.717 std::moneypunct< _CharT, _Intl > Class Template Reference	2405
6.717.1 Detailed Description	2407
6.717.2 Member Typedef Documentation	2407
6.717.3 Constructor & Destructor Documentation	2408
6.717.4 Member Function Documentation	2409
6.717.5 Member Data Documentation	2414
6.718 std::moneypunct_byname< _CharT, _Intl > Class Template Reference	2414
6.718.1 Detailed Description	2416
6.718.2 Member Function Documentation	2416
6.718.3 Member Data Documentation	2422
6.719 std::move_iterator< _Iterator > Class Template Reference	2422
6.719.1 Detailed Description	2423
6.720 std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> Class Template Reference	2423
6.720.1 Detailed Description	2424
6.720.2 Constructor & Destructor Documentation	2424
6.720.3 Member Function Documentation	2425
6.720.4 Friends And Related Symbol Documentation	2426
6.721 std::__debug::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference	2427
6.721.1 Detailed Description	2429
6.722 std::multimap< _Key, _Tp, _Compare, _Alloc > Class Template Reference	2429

6.722.1 Detailed Description	2433
6.722.2 Constructor & Destructor Documentation	2434
6.722.3 Member Function Documentation	2437
6.723 std::multiplies< _Tp > Struct Template Reference	2455
6.723.1 Detailed Description	2455
6.723.2 Member Typedef Documentation	2455
6.724 std::multiplies< void > Struct Reference	2456
6.724.1 Detailed Description	2456
6.725 std::__debug::multiset< _Key, _Compare, _Allocator > Class Template Reference	2456
6.725.1 Detailed Description	2458
6.726 std::multiset< _Key, _Compare, _Alloc > Class Template Reference	2458
6.726.1 Detailed Description	2461
6.726.2 Constructor & Destructor Documentation	2462
6.726.3 Member Function Documentation	2464
6.727 __gnu_parallel::multiway_mergesort_exact_tag Struct Reference	2480
6.727.1 Detailed Description	2481
6.727.2 Member Function Documentation	2481
6.728 __gnu_parallel::multiway_mergesort_sampling_tag Struct Reference	2482
6.728.1 Detailed Description	2482
6.728.2 Member Function Documentation	2482
6.729 __gnu_parallel::multiway_mergesort_tag Struct Reference	2483
6.729.1 Detailed Description	2483
6.729.2 Member Function Documentation	2483
6.730 std::mutex Class Reference	2484
6.730.1 Detailed Description	2484
6.731 std::negate< _Tp > Struct Template Reference	2484
6.731.1 Detailed Description	2485
6.731.2 Member Typedef Documentation	2485
6.732 std::negate< void > Struct Reference	2485
6.732.1 Detailed Description	2485
6.733 std::negative_binomial_distribution< _IntType > Class Template Reference	2485
6.733.1 Detailed Description	2486
6.733.2 Member Typedef Documentation	2486
6.733.3 Member Function Documentation	2487
6.733.4 Friends And Related Symbol Documentation	2488
6.734 std::nested_exception Class Reference	2489
6.734.1 Detailed Description	2489
6.735 __gnu_cxx::limit_condition::never_adjustor Struct Reference	2489
6.735.1 Detailed Description	2489

6.736 __gnu_cxx::random_condition::never_adjustor Struct Reference	2489
6.736.1 Detailed Description	2489
6.737 __gnu_cxx::new_allocator< _Tp > Class Template Reference	2490
6.737.1 Detailed Description	2490
6.738 __gnu_pbds::detail::no_throw_copies< Key, Mapped > Struct Template Reference	2491
6.738.1 Detailed Description	2491
6.739 __gnu_pbds::detail::no_throw_copies< Key, null_type > Struct Template Reference	2491
6.739.1 Detailed Description	2491
6.740 std::normal_distribution< _RealType > Class Template Reference	2491
6.740.1 Detailed Description	2492
6.740.2 Member Typedef Documentation	2492
6.740.3 Constructor & Destructor Documentation	2493
6.740.4 Member Function Documentation	2493
6.740.5 Friends And Related Symbol Documentation	2494
6.741 std::nostopstate_t Struct Reference	2495
6.741.1 Detailed Description	2495
6.742 std::not_equal_to< _Tp > Struct Template Reference	2495
6.742.1 Detailed Description	2496
6.742.2 Member Typedef Documentation	2496
6.743 std::ranges::not_equal_to Struct Reference	2497
6.743.1 Detailed Description	2497
6.744 std::not_equal_to< void > Struct Reference	2497
6.744.1 Detailed Description	2497
6.745 __gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 > Struct Template Reference	2497
6.745.1 Detailed Description	2498
6.746 __gnu_pbds::null_type Struct Reference	2498
6.746.1 Detailed Description	2498
6.747 std::experimental::fundamentals_v1::nullopt_t Struct Reference	2499
6.747.1 Detailed Description	2499
6.748 std::nullopt_t Struct Reference	2499
6.748.1 Detailed Description	2499
6.749 std::num_get< _CharT, _InIter > Class Template Reference	2499
6.749.1 Detailed Description	2501
6.749.2 Member Typedef Documentation	2502
6.749.3 Constructor & Destructor Documentation	2502
6.749.4 Member Function Documentation	2502
6.749.5 Member Data Documentation	2515
6.750 std::num_put< _CharT, _OutIter > Class Template Reference	2515
6.750.1 Detailed Description	2517

6.750.2 Member Typedef Documentation	2517
6.750.3 Constructor & Destructor Documentation	2518
6.750.4 Member Function Documentation	2518
6.750.5 Member Data Documentation	2528
6.751 std::numeric_limits< _Tp > Struct Template Reference	2528
6.751.1 Detailed Description	2529
6.751.2 Member Function Documentation	2529
6.751.3 Member Data Documentation	2531
6.752 std::numeric_limits< bool > Struct Reference	2534
6.752.1 Detailed Description	2534
6.753 std::numeric_limits< char > Struct Reference	2535
6.753.1 Detailed Description	2535
6.754 std::numeric_limits< char16_t > Struct Reference	2536
6.754.1 Detailed Description	2536
6.755 std::numeric_limits< char32_t > Struct Reference	2537
6.755.1 Detailed Description	2537
6.756 std::numeric_limits< double > Struct Reference	2538
6.756.1 Detailed Description	2538
6.757 std::numeric_limits< float > Struct Reference	2539
6.757.1 Detailed Description	2539
6.758 std::numeric_limits< int > Struct Reference	2540
6.758.1 Detailed Description	2540
6.759 std::numeric_limits< long > Struct Reference	2541
6.759.1 Detailed Description	2541
6.760 std::numeric_limits< long double > Struct Reference	2542
6.760.1 Detailed Description	2542
6.761 std::numeric_limits< long long > Struct Reference	2543
6.761.1 Detailed Description	2543
6.762 std::numeric_limits< short > Struct Reference	2544
6.762.1 Detailed Description	2544
6.763 std::numeric_limits< signed char > Struct Reference	2545
6.763.1 Detailed Description	2545
6.764 std::numeric_limits< unsigned char > Struct Reference	2546
6.764.1 Detailed Description	2546
6.765 std::numeric_limits< unsigned int > Struct Reference	2547
6.765.1 Detailed Description	2547
6.766 std::numeric_limits< unsigned long > Struct Reference	2548
6.766.1 Detailed Description	2548
6.767 std::numeric_limits< unsigned long long > Struct Reference	2549

6.767.1 Detailed Description	2549
6.768 std::numeric_limits< unsigned short > Struct Reference	2550
6.768.1 Detailed Description	2550
6.769 std::numeric_limits< wchar_t > Struct Reference	2551
6.769.1 Detailed Description	2551
6.770 std::numpunct< _CharT > Class Template Reference	2552
6.770.1 Detailed Description	2553
6.770.2 Member Typedef Documentation	2553
6.770.3 Constructor & Destructor Documentation	2553
6.770.4 Member Function Documentation	2554
6.770.5 Member Data Documentation	2557
6.771 std::numpunct_byname< _CharT > Class Template Reference	2557
6.771.1 Detailed Description	2558
6.771.2 Member Function Documentation	2558
6.771.3 Member Data Documentation	2561
6.772 __gnu_parallel::omp_loop_static_tag Struct Reference	2561
6.772.1 Detailed Description	2562
6.772.2 Member Function Documentation	2562
6.773 __gnu_parallel::omp_loop_tag Struct Reference	2562
6.773.1 Detailed Description	2563
6.773.2 Member Function Documentation	2563
6.774 std::once_flag Struct Reference	2563
6.774.1 Detailed Description	2563
6.774.2 Constructor & Destructor Documentation	2563
6.774.3 Member Function Documentation	2564
6.774.4 Friends And Related Symbol Documentation	2564
6.775 std::experimental::fundamentals_v1::optional< _Tp > Class Template Reference	2564
6.775.1 Detailed Description	2566
6.776 std::optional< _Tp > Class Template Reference	2566
6.776.1 Detailed Description	2568
6.777 std::ostream_iterator< _Tp, _CharT, _Traits > Class Template Reference	2568
6.777.1 Detailed Description	2569
6.777.2 Member Typedef Documentation	2569
6.777.3 Constructor & Destructor Documentation	2570
6.777.4 Member Function Documentation	2570
6.778 std::experimental::fundamentals_v2::ostream_joiner< _DelimT, _CharT, _Traits > Class Template Reference	2571
6.778.1 Detailed Description	2571
6.779 std::ostreambuf_iterator< _CharT, _Traits > Class Template Reference	2571

6.779.1 Detailed Description	2572
6.779.2 Member Typedef Documentation	2573
6.779.3 Constructor & Destructor Documentation	2574
6.779.4 Member Function Documentation	2574
6.780 std::out_of_range Class Reference	2575
6.780.1 Detailed Description	2575
6.780.2 Member Function Documentation	2575
6.781 std::output_iterator_tag Struct Reference	2575
6.781.1 Detailed Description	2576
6.782 __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > Class Template Reference	2576
6.782.1 Detailed Description	2578
6.782.2 Member Function Documentation	2578
6.783 __gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc > Class Template Reference	2579
6.783.1 Detailed Description	2580
6.783.2 Member Function Documentation	2580
6.784 __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc > Class Template Reference	2580
6.784.1 Detailed Description	2582
6.784.2 Member Function Documentation	2582
6.785 __gnu_pbds::ov_tree_tag Struct Reference	2582
6.785.1 Detailed Description	2583
6.786 std::overflow_error Class Reference	2584
6.786.1 Detailed Description	2584
6.786.2 Member Function Documentation	2584
6.787 std::owner_less< _Tp > Struct Template Reference	2585
6.787.1 Detailed Description	2585
6.788 std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > > Struct Template Reference	2585
6.788.1 Detailed Description	2585
6.788.2 Member Typedef Documentation	2585
6.789 std::owner_less< shared_ptr< _Tp > > Struct Template Reference	2586
6.789.1 Detailed Description	2586
6.789.2 Member Typedef Documentation	2586
6.790 std::owner_less< void > Struct Reference	2586
6.790.1 Detailed Description	2587
6.790.2 Member Typedef Documentation	2587
6.791 std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > > Struct Template Reference	2587
6.791.1 Detailed Description	2587
6.791.2 Member Typedef Documentation	2588
6.792 std::owner_less< weak_ptr< _Tp > > Struct Template Reference	2588

6.792.1 Detailed Description	2588
6.792.2 Member Typedef Documentation	2588
6.793 std::packaged_task< _Res(_ArgTypes...)> Class Template Reference	2589
6.793.1 Detailed Description	2589
6.794 std::pair< _T1, _T2 > Struct Template Reference	2589
6.794.1 Detailed Description	2591
6.794.2 Member Typedef Documentation	2592
6.794.3 Constructor & Destructor Documentation	2592
6.794.4 Member Function Documentation	2593
6.794.5 Member Data Documentation	2593
6.795 __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	2593
6.795.1 Detailed Description	2595
6.796 __gnu_pbds::pairing_heap_tag Struct Reference	2595
6.796.1 Detailed Description	2596
6.797 __gnu_parallel::parallel_tag Struct Reference	2596
6.797.1 Detailed Description	2597
6.797.2 Constructor & Destructor Documentation	2598
6.797.3 Member Function Documentation	2598
6.798 std::bernoulli_distribution::param_type Struct Reference	2598
6.798.1 Detailed Description	2599
6.799 std::binomial_distribution< _IntType >::param_type Struct Reference	2599
6.799.1 Detailed Description	2599
6.800 std::cauchy_distribution< _RealType >::param_type Struct Reference	2599
6.800.1 Detailed Description	2600
6.801 std::chi_squared_distribution< _RealType >::param_type Struct Reference	2600
6.801.1 Detailed Description	2600
6.802 std::discrete_distribution< _IntType >::param_type Struct Reference	2600
6.802.1 Detailed Description	2601
6.803 std::exponential_distribution< _RealType >::param_type Struct Reference	2601
6.803.1 Detailed Description	2601
6.804 std::extreme_value_distribution< _RealType >::param_type Struct Reference	2601
6.804.1 Detailed Description	2602
6.805 std::fisher_f_distribution< _RealType >::param_type Struct Reference	2602
6.805.1 Detailed Description	2602
6.806 std::gamma_distribution< _RealType >::param_type Struct Reference	2602
6.806.1 Detailed Description	2603
6.807 std::geometric_distribution< _IntType >::param_type Struct Reference	2603
6.807.1 Detailed Description	2603
6.808 std::lognormal_distribution< _RealType >::param_type Struct Reference	2603

6.808.1 Detailed Description	2604
6.809 std::negative_binomial_distribution< _IntType >::param_type Struct Reference	2604
6.809.1 Detailed Description	2604
6.810 std::normal_distribution< _RealType >::param_type Struct Reference	2604
6.810.1 Detailed Description	2605
6.811 std::piecewise_constant_distribution< _RealType >::param_type Struct Reference	2605
6.811.1 Detailed Description	2606
6.812 std::piecewise_linear_distribution< _RealType >::param_type Struct Reference	2606
6.812.1 Detailed Description	2606
6.813 std::poisson_distribution< _IntType >::param_type Struct Reference	2606
6.813.1 Detailed Description	2607
6.814 std::student_t_distribution< _RealType >::param_type Struct Reference	2607
6.814.1 Detailed Description	2607
6.815 std::uniform_int_distribution< _IntType >::param_type Struct Reference	2607
6.815.1 Detailed Description	2608
6.816 std::uniform_real_distribution< _RealType >::param_type Struct Reference	2608
6.816.1 Detailed Description	2608
6.817 std::weibull_distribution< _RealType >::param_type Struct Reference	2608
6.817.1 Detailed Description	2609
6.818 __gnu_pbds::detail::pat_trie_base Struct Reference	2609
6.818.1 Detailed Description	2610
6.818.2 Member Enumeration Documentation	2610
6.819 __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc > Class Template Reference	2610
6.819.1 Detailed Description	2612
6.819.2 Member Enumeration Documentation	2612
6.819.3 Member Function Documentation	2612
6.820 __gnu_pbds::pat_trie_tag Struct Reference	2613
6.820.1 Detailed Description	2613
6.821 std::experimental::filesystem::v1::path Class Reference	2614
6.821.1 Detailed Description	2616
6.822 std::filesystem::path Class Reference	2616
6.822.1 Detailed Description	2618
6.822.2 Member Enumeration Documentation	2619
6.822.3 Friends And Related Symbol Documentation	2619
6.823 std::piecewise_constant_distribution< _RealType > Class Template Reference	2620
6.823.1 Detailed Description	2621
6.823.2 Member Typedef Documentation	2621
6.823.3 Member Function Documentation	2621

6.823.4 Friends And Related Symbol Documentation	2622
6.824 std::piecewise_construct_t Struct Reference	2623
6.824.1 Detailed Description	2623
6.825 std::piecewise_linear_distribution<_RealType> Class Template Reference	2623
6.825.1 Detailed Description	2624
6.825.2 Member Typedef Documentation	2625
6.825.3 Member Function Documentation	2625
6.825.4 Friends And Related Symbol Documentation	2626
6.826 std::plus<_Tp> Struct Template Reference	2627
6.826.1 Detailed Description	2627
6.826.2 Member Typedef Documentation	2628
6.826.3 Member Function Documentation	2628
6.827 __gnu_pbds::point_invalidation_guarantee Struct Reference	2628
6.827.1 Detailed Description	2629
6.828 std::pointer_to_binary_function<_Arg1, _Arg2, _Result> Class Template Reference	2629
6.828.1 Detailed Description	2629
6.828.2 Member Typedef Documentation	2630
6.829 std::pointer_to_unary_function<_Arg, _Result> Class Template Reference	2630
6.829.1 Detailed Description	2631
6.829.2 Member Typedef Documentation	2631
6.830 std::pointer_traits<_Ptr> Struct Template Reference	2631
6.830.1 Detailed Description	2631
6.830.2 Member Typedef Documentation	2632
6.830.3 Member Function Documentation	2632
6.831 std::pointer_traits<_Tp*> Struct Template Reference	2633
6.831.1 Detailed Description	2633
6.831.2 Member Typedef Documentation	2633
6.831.3 Member Function Documentation	2634
6.832 std::poisson_distribution<_IntType> Class Template Reference	2634
6.832.1 Detailed Description	2635
6.832.2 Member Typedef Documentation	2635
6.832.3 Member Function Documentation	2635
6.832.4 Friends And Related Symbol Documentation	2636
6.833 std::pmr::pool_options Struct Reference	2637
6.833.1 Detailed Description	2638
6.833.2 Member Data Documentation	2638
6.834 __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc> Class Template Reference	2638
6.834.1 Detailed Description	2639
6.834.2 Constructor & Destructor Documentation	2640

6.835 std::priority_queue< _Tp, _Sequence, _Compare > Class Template Reference	2640
6.835.1 Detailed Description	2642
6.835.2 Constructor & Destructor Documentation	2642
6.835.3 Member Function Documentation	2643
6.836 __gnu_pbds::priority_queue_tag Struct Reference	2644
6.836.1 Detailed Description	2644
6.837 __gnu_pbds::detail::probe_fn_base< _Alloc > Class Template Reference	2644
6.837.1 Detailed Description	2645
6.838 __gnu_cxx::project1st< _Arg1, _Arg2 > Struct Template Reference	2645
6.838.1 Detailed Description	2645
6.838.2 Member Typedef Documentation	2645
6.839 __gnu_cxx::project2nd< _Arg1, _Arg2 > Struct Template Reference	2645
6.839.1 Detailed Description	2646
6.839.2 Member Typedef Documentation	2646
6.840 std::projected< _Iter, _Proj > Struct Template Reference	2646
6.840.1 Detailed Description	2646
6.841 std::promise< _Res > Class Template Reference	2647
6.841.1 Detailed Description	2647
6.842 std::promise< _Res & > Class Template Reference	2647
6.842.1 Detailed Description	2648
6.843 std::promise< void > Class Reference	2648
6.843.1 Detailed Description	2648
6.844 std::experimental::fundamentals_v2::propagate_const< _Tp > Class Template Reference	2648
6.844.1 Detailed Description	2649
6.845 __gnu_pbds::quadratic_probe_fn< Size_Type > Class Template Reference	2650
6.845.1 Detailed Description	2650
6.845.2 Member Function Documentation	2650
6.846 std::queue< _Tp, _Sequence > Class Template Reference	2650
6.846.1 Detailed Description	2651
6.846.2 Constructor & Destructor Documentation	2652
6.846.3 Member Function Documentation	2652
6.846.4 Member Data Documentation	2653
6.847 __gnu_parallel::quicksort_tag Struct Reference	2653
6.847.1 Detailed Description	2654
6.847.2 Member Function Documentation	2654
6.848 std::random_access_iterator_tag Struct Reference	2655
6.848.1 Detailed Description	2655
6.849 __gnu_cxx::random_condition Struct Reference	2655
6.849.1 Detailed Description	2656

6.850 <code>std::random_device</code> Class Reference	2656
6.850.1 Detailed Description	2656
6.850.2 Member Typedef Documentation	2656
6.851 <code>std::range_error</code> Class Reference	2657
6.851.1 Detailed Description	2657
6.851.2 Member Function Documentation	2657
6.852 <code>__gnu_pbds::range_invalidation_guarantee</code> Struct Reference	2657
6.852.1 Detailed Description	2658
6.853 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash > Class</code> Template Reference	2658
6.853.1 Detailed Description	2659
6.854 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false > Class Template</code> Reference	2659
6.854.1 Detailed Description	2659
6.855 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true > Class Template</code> Reference	2659
6.855.1 Detailed Description	2660
6.856 <code>__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false > Class Template</code> Reference	2660
6.856.1 Detailed Description	2660
6.857 <code>__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true > Class Template</code> Reference	2660
6.857.1 Detailed Description	2661
6.858 <code>__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_↵</code> <code>Hash > Class Template Reference</code>	2661
6.858.1 Detailed Description	2661
6.859 <code>__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false ></code> Class Template Reference	2661
6.859.1 Detailed Description	2662
6.860 <code>__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true ></code> Class Template Reference	2662
6.860.1 Detailed Description	2663
6.861 <code>__gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false ></code> Class Template Reference	2663
6.861.1 Detailed Description	2663
6.862 <code>std::rank< typename > Struct Template Reference</code>	2664
6.862.1 Detailed Description	2664
6.863 <code>std::ratio< _Num, _Den > Struct Template Reference</code>	2664
6.863.1 Detailed Description	2665
6.864 <code>std::ratio_equal< _R1, _R2 > Struct Template Reference</code>	2665
6.864.1 Detailed Description	2665
6.865 <code>std::ratio_greater< _R1, _R2 > Struct Template Reference</code>	2666

6.865.1 Detailed Description	2666
6.866 std::ratio_greater_equal< _R1, _R2 > Struct Template Reference	2667
6.866.1 Detailed Description	2667
6.867 std::ratio_less< _R1, _R2 > Struct Template Reference	2667
6.867.1 Detailed Description	2667
6.868 std::ratio_less_equal< _R1, _R2 > Struct Template Reference	2668
6.868.1 Detailed Description	2668
6.869 std::ratio_not_equal< _R1, _R2 > Struct Template Reference	2669
6.869.1 Detailed Description	2669
6.870 std::raw_storage_iterator< _OutputIterator, _Tp > Class Template Reference	2669
6.870.1 Detailed Description	2670
6.870.2 Member Typedef Documentation	2670
6.871 __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc > Struct Template Reference	2671
6.871.1 Detailed Description	2675
6.871.2 Member Function Documentation	2675
6.872 __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > Class Template Reference	2676
6.872.1 Detailed Description	2679
6.872.2 Member Function Documentation	2679
6.873 __gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc > Struct Template Reference	2680
6.873.1 Detailed Description	2680
6.874 __gnu_pbds::rb_tree_tag Struct Reference	2680
6.874.1 Detailed Description	2681
6.875 __gnu_pbds::detail::rc< _Node, _Alloc > Class Template Reference	2681
6.875.1 Detailed Description	2682
6.876 __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	2682
6.876.1 Detailed Description	2684
6.877 __gnu_pbds::rc_binomial_heap_tag Struct Reference	2684
6.877.1 Detailed Description	2684
6.878 __gnu_pbds::detail::rebind_traits< _Alloc, T > Struct Template Reference	2684
6.878.1 Detailed Description	2685
6.879 std::filesystem::recursive_directory_iterator Class Reference	2685
6.879.1 Detailed Description	2686
6.880 __gnu_cxx::recursive_init_error Class Reference	2686
6.880.1 Detailed Description	2686
6.880.2 Member Function Documentation	2686
6.881 std::recursive_mutex Class Reference	2687
6.881.1 Detailed Description	2687
6.882 std::recursive_timed_mutex Class Reference	2687

6.882.1 Detailed Description	2687
6.883 std::bitset< _Nb >::reference Class Reference	2687
6.883.1 Detailed Description	2688
6.884 std::tr2::dynamic_bitset< _WordT, _Alloc >::reference Class Reference	2688
6.884.1 Detailed Description	2688
6.885 std::reference_wrapper< _Tp > Class Template Reference	2688
6.885.1 Detailed Description	2689
6.885.2 Friends And Related Symbol Documentation	2689
6.886 std::regex_error Class Reference	2690
6.886.1 Detailed Description	2690
6.886.2 Constructor & Destructor Documentation	2691
6.886.3 Member Function Documentation	2691
6.887 std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference	2691
6.887.1 Detailed Description	2692
6.887.2 Constructor & Destructor Documentation	2692
6.887.3 Member Function Documentation	2693
6.888 std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference	2694
6.888.1 Detailed Description	2694
6.888.2 Constructor & Destructor Documentation	2695
6.888.3 Member Function Documentation	2697
6.889 std::regex_traits< _Ch_type > Class Template Reference	2698
6.889.1 Detailed Description	2699
6.889.2 Constructor & Destructor Documentation	2699
6.889.3 Member Function Documentation	2699
6.890 std::remove_all_extents< _Tp > Struct Template Reference	2704
6.890.1 Detailed Description	2704
6.891 std::remove_const< _Tp > Struct Template Reference	2704
6.891.1 Detailed Description	2704
6.892 std::remove_cv< _Tp > Struct Template Reference	2704
6.892.1 Detailed Description	2704
6.893 std::remove_extent< _Tp > Struct Template Reference	2705
6.893.1 Detailed Description	2705
6.894 std::remove_pointer< _Tp > Struct Template Reference	2705
6.894.1 Detailed Description	2705
6.895 std::remove_reference< _Tp > Struct Template Reference	2705
6.895.1 Detailed Description	2705
6.896 std::remove_volatile< _Tp > Struct Template Reference	2705
6.896.1 Detailed Description	2706
6.897 __gnu_pbds::resize_error Struct Reference	2706

6.897.1 Detailed Description	2706
6.897.2 Member Function Documentation	2706
6.898 __gnu_pbds::detail::resize_policy<_Tp> Class Template Reference	2707
6.898.1 Detailed Description	2707
6.899 std::result_of<_Signature> Struct Template Reference	2707
6.899.1 Detailed Description	2707
6.900 std::reverse_iterator<_Iterator> Class Template Reference	2708
6.900.1 Detailed Description	2709
6.900.2 Constructor & Destructor Documentation	2709
6.900.3 Member Function Documentation	2710
6.901 __gnu_cxx::rope<_CharT, _Alloc> Class Template Reference	2712
6.901.1 Detailed Description	2717
6.902 std::runtime_error Class Reference	2717
6.902.1 Detailed Description	2718
6.902.2 Constructor & Destructor Documentation	2718
6.902.3 Member Function Documentation	2718
6.903 __gnu_pbds::sample_probe_fn Class Reference	2718
6.903.1 Detailed Description	2718
6.903.2 Constructor & Destructor Documentation	2719
6.903.3 Member Function Documentation	2719
6.904 __gnu_pbds::sample_range_hashing Class Reference	2719
6.904.1 Detailed Description	2719
6.904.2 Member Typedef Documentation	2720
6.904.3 Constructor & Destructor Documentation	2720
6.904.4 Member Function Documentation	2720
6.905 __gnu_pbds::sample_ranged_hash_fn Class Reference	2720
6.905.1 Detailed Description	2721
6.905.2 Constructor & Destructor Documentation	2721
6.905.3 Member Function Documentation	2721
6.906 __gnu_pbds::sample_ranged_probe_fn Class Reference	2721
6.906.1 Detailed Description	2722
6.907 __gnu_pbds::sample_resize_policy Class Reference	2722
6.907.1 Detailed Description	2722
6.907.2 Member Typedef Documentation	2722
6.907.3 Constructor & Destructor Documentation	2723
6.907.4 Member Function Documentation	2723
6.908 __gnu_pbds::sample_resize_trigger Class Reference	2724
6.908.1 Detailed Description	2725
6.908.2 Member Typedef Documentation	2725

6.908.3 Constructor & Destructor Documentation	2725
6.908.4 Member Function Documentation	2725
6.909 <code>__gnu_pbds::sample_size_policy</code> Class Reference	2727
6.909.1 Detailed Description	2727
6.909.2 Member Typedef Documentation	2728
6.909.3 Constructor & Destructor Documentation	2728
6.909.4 Member Function Documentation	2728
6.910 <code>__gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc ></code> Class Template Reference	2728
6.910.1 Detailed Description	2728
6.911 <code>__gnu_pbds::sample_trie_access_traits</code> Struct Reference	2729
6.911.1 Detailed Description	2729
6.911.2 Member Typedef Documentation	2729
6.911.3 Member Function Documentation	2729
6.912 <code>__gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc ></code> Class Template Reference	2729
6.912.1 Detailed Description	2730
6.912.2 Constructor & Destructor Documentation	2730
6.912.3 Member Function Documentation	2730
6.913 <code>__gnu_pbds::sample_update_policy</code> Struct Reference	2730
6.913.1 Detailed Description	2731
6.913.2 Member Typedef Documentation	2731
6.913.3 Constructor & Destructor Documentation	2731
6.913.4 Member Function Documentation	2731
6.914 <code>__gnu_parallel::sampling_tag</code> Struct Reference	2731
6.914.1 Detailed Description	2732
6.914.2 Member Function Documentation	2732
6.915 <code>std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs ></code> Class Template Reference	2733
6.915.1 Detailed Description	2734
6.916 <code>std::scoped_lock< _MutexTypes ></code> Class Template Reference	2734
6.916.1 Detailed Description	2734
6.917 <code>std::seed_seq</code> Class Reference	2734
6.917.1 Detailed Description	2735
6.917.2 Member Typedef Documentation	2735
6.917.3 Constructor & Destructor Documentation	2735
6.918 <code>__gnu_cxx::select1st< _Pair ></code> Struct Template Reference	2735
6.918.1 Detailed Description	2736
6.918.2 Member Typedef Documentation	2736
6.919 <code>__gnu_cxx::select2nd< _Pair ></code> Struct Template Reference	2736
6.919.1 Detailed Description	2736

6.919.2 Member Typedef Documentation	2736
6.920 __gnu_pbds::detail::select_value_type< Key, Mapped > Struct Template Reference	2737
6.920.1 Detailed Description	2737
6.921 __gnu_pbds::detail::select_value_type< Key, null_type > Struct Template Reference	2737
6.921.1 Detailed Description	2737
6.922 std::basic_istream< _CharT, _Traits >::sentry Class Reference	2737
6.922.1 Detailed Description	2737
6.922.2 Member Typedef Documentation	2738
6.922.3 Constructor & Destructor Documentation	2738
6.922.4 Member Function Documentation	2738
6.923 std::basic_ostream< _CharT, _Traits >::sentry Class Reference	2739
6.923.1 Detailed Description	2739
6.923.2 Constructor & Destructor Documentation	2739
6.923.3 Member Function Documentation	2739
6.924 __gnu_pbds::sequence_tag Struct Reference	2740
6.924.1 Detailed Description	2740
6.925 __gnu_parallel::sequential_tag Struct Reference	2740
6.925.1 Detailed Description	2740
6.926 std::__debug::set< _Key, _Compare, _Allocator > Class Template Reference	2740
6.926.1 Detailed Description	2743
6.927 std::set< _Key, _Compare, _Alloc > Class Template Reference	2743
6.927.1 Detailed Description	2746
6.927.2 Member Typedef Documentation	2746
6.927.3 Constructor & Destructor Documentation	2748
6.927.4 Member Function Documentation	2751
6.928 std::shared_future< _Res > Class Template Reference	2766
6.928.1 Detailed Description	2767
6.928.2 Constructor & Destructor Documentation	2767
6.928.3 Member Function Documentation	2767
6.929 std::shared_future< _Res & > Class Template Reference	2768
6.929.1 Detailed Description	2768
6.929.2 Constructor & Destructor Documentation	2769
6.929.3 Member Function Documentation	2769
6.930 std::shared_future< void > Class Reference	2770
6.930.1 Detailed Description	2770
6.930.2 Constructor & Destructor Documentation	2770
6.930.3 Member Function Documentation	2771
6.931 std::shared_lock< _Mutex > Class Template Reference	2771
6.931.1 Detailed Description	2772

6.932 <code>std::shared_mutex</code> Class Reference	2772
6.932.1 Detailed Description	2772
6.933 <code>std::shared_ptr<_Tp></code> Class Template Reference	2772
6.933.1 Detailed Description	2776
6.933.2 Member Typedef Documentation	2776
6.933.3 Constructor & Destructor Documentation	2777
6.933.4 Member Function Documentation	2783
6.934 <code>std::shared_timed_mutex</code> Class Reference	2784
6.934.1 Detailed Description	2784
6.935 <code>std::shuffle_order_engine<_RandomNumberEngine, __k></code> Class Template Reference	2784
6.935.1 Detailed Description	2785
6.935.2 Member Typedef Documentation	2785
6.935.3 Constructor & Destructor Documentation	2785
6.935.4 Member Function Documentation	2787
6.935.5 Friends And Related Symbol Documentation	2788
6.936 <code>std::ranges::single_view<_Tp></code> Class Template Reference	2789
6.936.1 Detailed Description	2790
6.937 <code>std::slice</code> Class Reference	2790
6.937.1 Detailed Description	2790
6.937.2 Friends And Related Symbol Documentation	2790
6.938 <code>std::slice_array<_Tp></code> Class Template Reference	2791
6.938.1 Detailed Description	2792
6.938.2 Member Function Documentation	2792
6.939 <code>__gnu_cxx::slist<_Tp, _Alloc></code> Class Template Reference	2793
6.939.1 Detailed Description	2795
6.940 <code>std::source_location</code> Struct Reference	2795
6.940.1 Detailed Description	2795
6.941 <code>std::experimental::filesystem::v1::space_info</code> Struct Reference	2796
6.941.1 Detailed Description	2796
6.942 <code>std::filesystem::space_info</code> Struct Reference	2796
6.942.1 Detailed Description	2796
6.943 <code>__gnu_pbds::detail::splay_tree_map<Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc></code> Class Template Reference	2796
6.943.1 Detailed Description	2799
6.943.2 Member Function Documentation	2799
6.944 <code>__gnu_pbds::detail::splay_tree_node<Value_Type, Metadata, _Alloc></code> Struct Template Reference	2800
6.944.1 Detailed Description	2800
6.945 <code>__gnu_pbds::splay_tree_tag</code> Struct Reference	2800
6.945.1 Detailed Description	2801

6.946 std::stack< _Tp, _Sequence > Class Template Reference	2801
6.946.1 Detailed Description	2802
6.946.2 Constructor & Destructor Documentation	2803
6.946.3 Member Function Documentation	2803
6.947 __gnu_cxx::stdio_filebuf< _CharT, _Traits > Class Template Reference	2804
6.947.1 Detailed Description	2807
6.947.2 Constructor & Destructor Documentation	2807
6.947.3 Member Function Documentation	2808
6.947.4 Member Data Documentation	2822
6.948 __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits > Class Template Reference	2825
6.948.1 Detailed Description	2827
6.948.2 Member Function Documentation	2827
6.948.3 Member Data Documentation	2839
6.949 std::chrono::_V2::steady_clock Struct Reference	2840
6.949.1 Detailed Description	2840
6.950 std::stop_callback< _Callback > Class Template Reference	2840
6.950.1 Detailed Description	2840
6.951 std::stop_source Class Reference	2841
6.951.1 Detailed Description	2841
6.952 std::stop_token Class Reference	2841
6.952.1 Detailed Description	2841
6.953 __gnu_pbds::detail::stored_data< _Tv, _Th, Store_Hash > Struct Template Reference	2842
6.953.1 Detailed Description	2842
6.954 __gnu_pbds::detail::stored_data< _Tv, _Th, false > Struct Template Reference	2842
6.954.1 Detailed Description	2843
6.955 __gnu_pbds::detail::stored_hash< _Th > Struct Template Reference	2843
6.955.1 Detailed Description	2844
6.956 __gnu_pbds::detail::stored_value< _Tv > Struct Template Reference	2844
6.956.1 Detailed Description	2845
6.957 __gnu_pbds::string_tag Struct Reference	2845
6.957.1 Detailed Description	2846
6.958 std::student_t_distribution< _RealType > Class Template Reference	2846
6.958.1 Detailed Description	2847
6.958.2 Member Typedef Documentation	2847
6.958.3 Member Function Documentation	2847
6.958.4 Friends And Related Symbol Documentation	2848
6.959 std::sub_match< _Bilter > Class Template Reference	2849
6.959.1 Detailed Description	2853
6.959.2 Member Typedef Documentation	2853

6.959.3 Member Function Documentation	2853
6.959.4 Friends And Related Symbol Documentation	2855
6.959.5 Member Data Documentation	2857
6.960 std::ranges::subrange< _It, _Sent, _Kind > Class Template Reference	2857
6.960.1 Detailed Description	2858
6.961 std::subtract_with_carry_engine< _UIntType, __w, __s, __r > Class Template Reference	2859
6.961.1 Detailed Description	2859
6.961.2 Member Typedef Documentation	2860
6.961.3 Constructor & Destructor Documentation	2860
6.961.4 Member Function Documentation	2860
6.961.5 Friends And Related Symbol Documentation	2861
6.962 __gnu_cxx::subtractive_rng Class Reference	2863
6.962.1 Detailed Description	2863
6.962.2 Member Typedef Documentation	2863
6.962.3 Constructor & Destructor Documentation	2863
6.962.4 Member Function Documentation	2864
6.963 std::pmr::synchronized_pool_resource Class Reference	2864
6.963.1 Detailed Description	2865
6.963.2 Member Function Documentation	2865
6.964 __gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits > Struct Template Reference	2865
6.964.1 Detailed Description	2866
6.965 std::chrono::_V2::system_clock Struct Reference	2866
6.965.1 Detailed Description	2866
6.966 std::system_error Class Reference	2866
6.966.1 Detailed Description	2867
6.966.2 Member Function Documentation	2867
6.967 __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp > Struct Template Reference	2867
6.967.1 Detailed Description	2868
6.967.2 Constructor & Destructor Documentation	2868
6.967.3 Member Function Documentation	2869
6.968 __gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	2869
6.968.1 Detailed Description	2871
6.969 __gnu_pbds::thin_heap_tag Struct Reference	2871
6.969.1 Detailed Description	2871
6.970 std::thread Class Reference	2872
6.970.1 Detailed Description	2872
6.970.2 Member Function Documentation	2873
6.971 __gnu_cxx::throw_allocator_base< _Tp, _Cond > Class Template Reference	2873
6.971.1 Detailed Description	2874

6.972 __gnu_cxx::throw_allocator_limit< _Tp > Struct Template Reference	2874
6.972.1 Detailed Description	2876
6.973 __gnu_cxx::throw_allocator_random< _Tp > Struct Template Reference	2876
6.973.1 Detailed Description	2877
6.974 __gnu_cxx::throw_value_base< _Cond > Struct Template Reference	2878
6.974.1 Detailed Description	2878
6.975 __gnu_cxx::throw_value_limit Struct Reference	2878
6.975.1 Detailed Description	2880
6.976 __gnu_cxx::throw_value_random Struct Reference	2880
6.976.1 Detailed Description	2881
6.977 std::time_base Class Reference	2881
6.977.1 Detailed Description	2881
6.978 std::time_get< _CharT, _InIter > Class Template Reference	2881
6.978.1 Detailed Description	2883
6.978.2 Member Typedef Documentation	2883
6.978.3 Constructor & Destructor Documentation	2884
6.978.4 Member Function Documentation	2884
6.978.5 Member Data Documentation	2892
6.979 std::time_get_byname< _CharT, _InIter > Class Template Reference	2893
6.979.1 Detailed Description	2894
6.979.2 Member Function Documentation	2894
6.979.3 Member Data Documentation	2903
6.980 std::chrono::time_point< _Clock, _Dur > Class Template Reference	2903
6.980.1 Detailed Description	2904
6.981 std::time_put< _CharT, _OutIter > Class Template Reference	2904
6.981.1 Detailed Description	2905
6.981.2 Member Typedef Documentation	2905
6.981.3 Constructor & Destructor Documentation	2905
6.981.4 Member Function Documentation	2906
6.981.5 Member Data Documentation	2907
6.982 std::time_put_byname< _CharT, _OutIter > Class Template Reference	2908
6.982.1 Detailed Description	2909
6.982.2 Member Function Documentation	2909
6.982.3 Member Data Documentation	2911
6.983 std::timed_mutex Class Reference	2911
6.983.1 Detailed Description	2911
6.984 std::to_chars_result Struct Reference	2911
6.984.1 Detailed Description	2911
6.985 std::chrono::treat_as_floating_point< _Rep > Struct Template Reference	2912

6.985.1 Detailed Description	2912
6.986 <code>__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc ></code> Class Template Reference	2912
6.986.1 Detailed Description	2913
6.986.2 Member Typedef Documentation	2914
6.986.3 Constructor & Destructor Documentation	2914
6.987 <code>__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp ></code> Struct Template Reference	2915
6.987.1 Detailed Description	2915
6.988 <code>__gnu_pbds::detail::tree_metadata_helper< Node_Update, false ></code> Struct Template Reference	2915
6.988.1 Detailed Description	2915
6.989 <code>__gnu_pbds::detail::tree_metadata_helper< Node_Update, true ></code> Struct Template Reference	2915
6.989.1 Detailed Description	2915
6.990 <code>__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc ></code> Struct Template Reference	2915
6.990.1 Detailed Description	2916
6.991 <code>__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc ></code> Class Template Reference	2916
6.991.1 Detailed Description	2917
6.991.2 Member Function Documentation	2917
6.992 <code>__gnu_pbds::tree_tag</code> Struct Reference	2918
6.992.1 Detailed Description	2918
6.993 <code>__gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc ></code> Struct Template Reference	2918
6.993.1 Detailed Description	2918
6.994 <code>__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc ></code> Struct Template Reference	2918
6.994.1 Detailed Description	2919
6.994.2 Member Typedef Documentation	2919
6.995 <code>__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc ></code> Struct Template Reference	2919
6.995.1 Detailed Description	2920
6.995.2 Member Typedef Documentation	2920
6.996 <code>__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc ></code> Struct Template Reference	2920
6.996.1 Detailed Description	2921
6.996.2 Member Typedef Documentation	2922
6.997 <code>__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc ></code> Struct Template Reference	2922
6.997.1 Detailed Description	2922
6.997.2 Member Typedef Documentation	2922
6.998 <code>__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc ></code> Struct Template Reference	2922

6.998.1 Detailed Description	2923
6.998.2 Member Typedef Documentation	2923
6.999 <code>__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc ></code> Struct Template Reference	2924
6.999.1 Detailed Description	2925
6.999.2 Member Typedef Documentation	2925
6.1000 <code>__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc ></code> Class Template Reference	2925
6.1000.1 Detailed Description	2926
6.1000.2 Member Typedef Documentation	2926
6.1000.3 Constructor & Destructor Documentation	2926
6.1001 <code>__gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTP ></code> Struct Template Reference	2927
6.1001.1 Detailed Description	2927
6.1002 <code>__gnu_pbds::detail::trie_metadata_helper< Node_Update, false ></code> Struct Template Reference	2927
6.1002.1 Detailed Description	2928
6.1003 <code>__gnu_pbds::detail::trie_metadata_helper< Node_Update, true ></code> Struct Template Reference	2928
6.1003.1 Detailed Description	2928
6.1004 <code>__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc ></code> Struct Template Reference	2928
6.1004.1 Detailed Description	2928
6.1005 <code>__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc ></code> Class Tem- plate Reference	2928
6.1005.1 Detailed Description	2930
6.1005.2 Member Function Documentation	2930
6.1006 <code>__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc ></code> Class Template Refer- ence	2931
6.1006.1 Detailed Description	2932
6.1006.2 Member Function Documentation	2932
6.1007 <code>__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc ></code> Class Tem- plate Reference	2932
6.1007.1 Detailed Description	2934
6.1007.2 Member Typedef Documentation	2934
6.1007.3 Member Function Documentation	2934
6.1008 <code>__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc ></code> Struct Template Reference	2935
6.1008.1 Detailed Description	2936
6.1008.2 Member Typedef Documentation	2936
6.1008.3 Member Function Documentation	2936
6.1009 <code>__gnu_pbds::trie_tag</code> Struct Reference	2937
6.1009.1 Detailed Description	2938
6.1010 <code>__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc ></code> Struct Template Ref- erence	2938

6.1010.1 Detailed Description	2938
6.1011 <code>__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc > Struct</code> Template Reference	2938
6.1011.1 Detailed Description	2939
6.1011.2 Member Typedef Documentation	2939
6.1012 <code>__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc > Struct</code> Template Reference	2940
6.1012.1 Detailed Description	2940
6.1012.2 Member Typedef Documentation	2940
6.1013 <code>__gnu_pbds::trivial_iterator_tag</code> Struct Reference	2941
6.1013.1 Detailed Description	2941
6.1014 <code>std::try_to_lock_t</code> Struct Reference	2941
6.1014.1 Detailed Description	2941
6.1015 <code>std::tuple< _Elements > Class</code> Template Reference	2941
6.1015.1 Detailed Description	2943
6.1016 <code>std::tuple< _T1, _T2 > Class</code> Template Reference	2943
6.1016.1 Detailed Description	2945
6.1017 <code>std::tuple_element< __i, _Tp > Struct</code> Template Reference	2945
6.1017.1 Detailed Description	2945
6.1018 <code>std::tuple_element< 0, pair< _Tp1, _Tp2 > > Struct</code> Template Reference	2945
6.1018.1 Detailed Description	2945
6.1019 <code>std::tuple_element< 1, pair< _Tp1, _Tp2 > > Struct</code> Template Reference	2946
6.1019.1 Detailed Description	2946
6.1020 <code>std::tuple_element< __i, tuple< _Types... > > Struct</code> Template Reference	2946
6.1020.1 Detailed Description	2946
6.1021 <code>std::tuple_element< _Ind, array< _Tp, _Nm > > Struct</code> Template Reference	2946
6.1021.1 Detailed Description	2946
6.1022 <code>std::tuple_size< _Tp > Struct</code> Template Reference	2946
6.1022.1 Detailed Description	2947
6.1023 <code>std::tuple_size< array< _Tp, _Nm > > Struct</code> Template Reference	2947
6.1023.1 Detailed Description	2947
6.1024 <code>std::tuple_size< pair< _Tp1, _Tp2 > > Struct</code> Template Reference	2948
6.1024.1 Detailed Description	2948
6.1025 <code>std::tuple_size< tuple< _Elements... > > Struct</code> Template Reference	2949
6.1025.1 Detailed Description	2949
6.1026 <code>__gnu_pbds::detail::entry_cmp< _VTP, Cmp_Fn, _Alloc, false >::type</code> Struct Reference	2949
6.1026.1 Detailed Description	2950
6.1027 <code>std::type_index</code> Struct Reference	2950
6.1027.1 Detailed Description	2950
6.1028 <code>std::type_info</code> Class Reference	2950

6.1028.1 Detailed Description	2951
6.1028.2 Constructor & Destructor Documentation	2951
6.1028.3 Member Function Documentation	2951
6.1029 __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference	2951
6.1029.1 Detailed Description	2952
6.1030 __gnu_cxx::unary_compose< _Operation1, _Operation2 > Class Template Reference	2953
6.1030.1 Detailed Description	2953
6.1030.2 Member Typedef Documentation	2953
6.1031 std::unary_function< _Arg, _Result > Struct Template Reference	2954
6.1031.1 Detailed Description	2954
6.1031.2 Member Typedef Documentation	2954
6.1032 std::unary_negate< _Predicate > Class Template Reference	2955
6.1032.1 Detailed Description	2955
6.1032.2 Member Typedef Documentation	2955
6.1033 __gnu_parallel::unbalanced_tag Struct Reference	2956
6.1033.1 Detailed Description	2956
6.1033.2 Member Function Documentation	2956
6.1034 std::underflow_error Class Reference	2957
6.1034.1 Detailed Description	2957
6.1034.2 Member Function Documentation	2957
6.1035 std::underlying_type< _Tp > Struct Template Reference	2957
6.1035.1 Detailed Description	2958
6.1036 std::uniform_int_distribution< _IntType > Class Template Reference	2958
6.1036.1 Detailed Description	2959
6.1036.2 Member Typedef Documentation	2959
6.1036.3 Constructor & Destructor Documentation	2959
6.1036.4 Member Function Documentation	2959
6.1036.5 Friends And Related Symbol Documentation	2960
6.1037 std::uniform_real_distribution< _RealType > Class Template Reference	2960
6.1037.1 Detailed Description	2961
6.1037.2 Member Typedef Documentation	2961
6.1037.3 Constructor & Destructor Documentation	2961
6.1037.4 Member Function Documentation	2962
6.1037.5 Friends And Related Symbol Documentation	2963
6.1038 std::unique_lock< _Mutex > Class Template Reference	2963
6.1038.1 Detailed Description	2964
6.1038.2 Friends And Related Symbol Documentation	2964
6.1039 std::unique_ptr< _Tp, _Dp > Class Template Reference	2964
6.1039.1 Detailed Description	2965

6.1039.2 Constructor & Destructor Documentation	2966
6.1039.3 Member Function Documentation	2967
6.1040 std::unique_ptr< _Tp[], _Dp > Class Template Reference	2970
6.1040.1 Detailed Description	2971
6.1040.2 Constructor & Destructor Documentation	2971
6.1040.3 Member Function Documentation	2972
6.1041 std::__debug::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	2975
6.1041.1 Detailed Description	2978
6.1042 std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	2978
6.1042.1 Detailed Description	2982
6.1042.2 Member Typedef Documentation	2983
6.1042.3 Constructor & Destructor Documentation	2985
6.1042.4 Member Function Documentation	2987
6.1043 std::__debug::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	3009
6.1043.1 Detailed Description	3011
6.1044 std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	3011
6.1044.1 Detailed Description	3015
6.1044.2 Member Typedef Documentation	3015
6.1044.3 Constructor & Destructor Documentation	3017
6.1044.4 Member Function Documentation	3019
6.1045 std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3036
6.1045.1 Detailed Description	3039
6.1046 std::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3039
6.1046.1 Detailed Description	3042
6.1046.2 Member Typedef Documentation	3042
6.1046.3 Constructor & Destructor Documentation	3044
6.1046.4 Member Function Documentation	3046
6.1047 std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3063
6.1047.1 Detailed Description	3066
6.1048 std::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3066
6.1048.1 Detailed Description	3069
6.1048.2 Member Typedef Documentation	3070
6.1048.3 Constructor & Destructor Documentation	3072
6.1048.4 Member Function Documentation	3074
6.1049 std::pmr::unsynchronized_pool_resource Class Reference	3091
6.1049.1 Detailed Description	3091
6.1049.2 Member Function Documentation	3091
6.1050 std::unwrap_ref_decay< _Tp > Struct Template Reference	3092
6.1050.1 Detailed Description	3092

6.1051 std::unwrap_reference< _Tp > Struct Template Reference	3092
6.1051.1 Detailed Description	3092
6.1052 std::uses_allocator< typename, typename > Struct Template Reference	3093
6.1052.1 Detailed Description	3093
6.1053 std::uses_allocator< tuple< _Types... >, _Alloc > Struct Template Reference	3093
6.1053.1 Detailed Description	3093
6.1054 std::valarray< _Tp > Class Template Reference	3094
6.1054.1 Detailed Description	3096
6.1054.2 Constructor & Destructor Documentation	3096
6.1054.3 Member Function Documentation	3096
6.1055 std::__debug::vector< _Tp, _Allocator > Class Template Reference	3099
6.1055.1 Detailed Description	3101
6.1055.2 Constructor & Destructor Documentation	3102
6.1056 std::vector< _Tp, _Alloc > Class Template Reference	3102
6.1056.1 Detailed Description	3105
6.1056.2 Constructor & Destructor Documentation	3105
6.1056.3 Member Function Documentation	3108
6.1057 std::vector< bool, _Alloc > Class Template Reference	3121
6.1057.1 Detailed Description	3124
6.1058 std::ranges::view_base Struct Reference	3124
6.1058.1 Detailed Description	3124
6.1059 std::ranges::view_interface< _Derived > Class Template Reference	3124
6.1059.1 Detailed Description	3125
6.1060 std::wbuffer_convert< _Codecv, _Elem, _Tr > Class Template Reference	3125
6.1060.1 Detailed Description	3127
6.1060.2 Member Typedef Documentation	3127
6.1060.3 Constructor & Destructor Documentation	3128
6.1060.4 Member Function Documentation	3129
6.1060.5 Member Data Documentation	3140
6.1061 std::weak_ptr< _Tp > Class Template Reference	3141
6.1061.1 Detailed Description	3142
6.1062 std::weibull_distribution< _RealType > Class Template Reference	3142
6.1062.1 Detailed Description	3143
6.1062.2 Member Typedef Documentation	3143
6.1062.3 Member Function Documentation	3143
6.1062.4 Friends And Related Symbol Documentation	3145
6.1063 std::wstring_convert< _Codecv, _Elem, _Wide_alloc, _Byte_alloc > Class Template Reference	3145
6.1063.1 Detailed Description	3145
6.1063.2 Constructor & Destructor Documentation	3146

6.1063.3 Member Function Documentation	3147
7 File Documentation	3148
7.1 compare File Reference	3148
7.1.1 Detailed Description	3150
7.2 cxxabi.h File Reference	3150
7.2.1 Detailed Description	3151
7.2.2 Function Documentation	3151
7.3 exception File Reference	3152
7.3.1 Detailed Description	3152
7.4 initializer_list File Reference	3152
7.4.1 Detailed Description	3152
7.5 new File Reference	3153
7.5.1 Detailed Description	3154
7.5.2 Function Documentation	3154
7.6 typeinfo File Reference	3154
7.6.1 Detailed Description	3155
7.7 algorithm File Reference	3155
7.7.1 Detailed Description	3155
7.8 algorithm File Reference	3155
7.8.1 Detailed Description	3155
7.9 algorithm File Reference	3155
7.9.1 Detailed Description	3156
7.10 algorithm File Reference	3157
7.10.1 Detailed Description	3157
7.11 any File Reference	3157
7.11.1 Detailed Description	3157
7.12 any File Reference	3158
7.12.1 Detailed Description	3158
7.13 array File Reference	3158
7.13.1 Detailed Description	3159
7.14 array File Reference	3160
7.14.1 Detailed Description	3160
7.15 atomic File Reference	3160
7.15.1 Detailed Description	3164
7.16 auto_ptr.h File Reference	3164
7.16.1 Detailed Description	3164
7.17 backward_warning.h File Reference	3165
7.17.1 Detailed Description	3165

7.18 binders.h File Reference	3165
7.18.1 Detailed Description	3165
7.19 hash_fun.h File Reference	3165
7.19.1 Detailed Description	3165
7.20 hash_map File Reference	3165
7.20.1 Detailed Description	3166
7.21 hash_set File Reference	3166
7.21.1 Detailed Description	3167
7.22 strstream File Reference	3167
7.22.1 Detailed Description	3167
7.23 barrier File Reference	3167
7.23.1 Detailed Description	3167
7.24 bit File Reference	3167
7.24.1 Detailed Description	3168
7.25 algorithmfwd.h File Reference	3168
7.25.1 Detailed Description	3174
7.26 algorithmfwd.h File Reference	3174
7.26.1 Detailed Description	3182
7.27 align.h File Reference	3182
7.27.1 Detailed Description	3182
7.28 alloc_traits.h File Reference	3182
7.28.1 Detailed Description	3183
7.29 alloc_traits.h File Reference	3183
7.29.1 Detailed Description	3183
7.30 allocated_ptr.h File Reference	3183
7.30.1 Detailed Description	3183
7.31 allocator.h File Reference	3184
7.31.1 Detailed Description	3184
7.32 atomic_base.h File Reference	3184
7.32.1 Detailed Description	3186
7.33 atomic_futex.h File Reference	3186
7.33.1 Detailed Description	3186
7.34 atomic_lockfree_defines.h File Reference	3186
7.34.1 Detailed Description	3187
7.35 atomic_timed_wait.h File Reference	3187
7.35.1 Detailed Description	3187
7.36 atomic_wait.h File Reference	3188
7.36.1 Detailed Description	3188
7.37 basic_ios.h File Reference	3188

7.37.1 Detailed Description	3189
7.38 <code>basic_ios.tcc</code> File Reference	3189
7.38.1 Detailed Description	3189
7.39 <code>basic_string.h</code> File Reference	3189
7.39.1 Detailed Description	3192
7.40 <code>basic_string.tcc</code> File Reference	3192
7.40.1 Detailed Description	3193
7.41 <code>boost_concept_check.h</code> File Reference	3193
7.41.1 Detailed Description	3193
7.42 <code>c++0x_warning.h</code> File Reference	3193
7.42.1 Detailed Description	3193
7.43 <code>char_traits.h</code> File Reference	3193
7.43.1 Detailed Description	3194
7.44 <code>charconv.h</code> File Reference	3194
7.44.1 Detailed Description	3194
7.45 <code>chrono.h</code> File Reference	3194
7.45.1 Detailed Description	3198
7.46 <code>codecvt.h</code> File Reference	3198
7.46.1 Detailed Description	3198
7.47 <code>concept_check.h</code> File Reference	3198
7.47.1 Detailed Description	3198
7.48 <code>cow_string.h</code> File Reference	3198
7.48.1 Detailed Description	3199
7.49 <code>cpp_type_traits.h</code> File Reference	3199
7.49.1 Detailed Description	3199
7.50 <code>cxxabi_forced.h</code> File Reference	3199
7.50.1 Detailed Description	3199
7.51 <code>cxxabi_init_exception.h</code> File Reference	3199
7.51.1 Detailed Description	3200
7.52 <code>deque.tcc</code> File Reference	3200
7.52.1 Detailed Description	3201
7.53 <code>enable_special_members.h</code> File Reference	3201
7.53.1 Detailed Description	3201
7.54 <code>erase_if.h</code> File Reference	3201
7.54.1 Detailed Description	3202
7.55 <code>exception.h</code> File Reference	3202
7.55.1 Detailed Description	3202
7.56 <code>exception_defines.h</code> File Reference	3202
7.56.1 Detailed Description	3202

7.57 exception_ptr.h File Reference	3202
7.57.1 Detailed Description	3203
7.58 forward_list.h File Reference	3203
7.58.1 Detailed Description	3203
7.59 forward_list.tcc File Reference	3203
7.59.1 Detailed Description	3204
7.60 fs_dir.h File Reference	3204
7.60.1 Detailed Description	3204
7.61 fs_dir.h File Reference	3204
7.61.1 Detailed Description	3205
7.62 fs_fwd.h File Reference	3205
7.62.1 Detailed Description	3206
7.63 fs_fwd.h File Reference	3207
7.63.1 Detailed Description	3208
7.64 fs_ops.h File Reference	3208
7.64.1 Detailed Description	3210
7.65 fs_ops.h File Reference	3211
7.65.1 Detailed Description	3213
7.66 fs_path.h File Reference	3213
7.66.1 Detailed Description	3213
7.67 fs_path.h File Reference	3213
7.67.1 Detailed Description	3214
7.68 fstream.tcc File Reference	3214
7.68.1 Detailed Description	3214
7.69 funtexcept.h File Reference	3214
7.69.1 Detailed Description	3214
7.70 functional_hash.h File Reference	3215
7.70.1 Detailed Description	3215
7.71 gslice.h File Reference	3215
7.71.1 Detailed Description	3215
7.72 gslice_array.h File Reference	3216
7.72.1 Detailed Description	3216
7.73 hash_bytes.h File Reference	3216
7.73.1 Detailed Description	3216
7.74 hashtable.h File Reference	3216
7.74.1 Detailed Description	3216
7.75 hashtable.h File Reference	3217
7.75.1 Detailed Description	3217
7.76 hashtable_policy.h File Reference	3217

7.76.1 Detailed Description	3217
7.77 indirect_array.h File Reference	3217
7.77.1 Detailed Description	3217
7.78 invoke.h File Reference	3217
7.78.1 Detailed Description	3218
7.79 ios_base.h File Reference	3218
7.79.1 Detailed Description	3219
7.80 istream.tcc File Reference	3219
7.80.1 Detailed Description	3220
7.81 iterator_concepts.h File Reference	3220
7.81.1 Detailed Description	3221
7.82 list.tcc File Reference	3221
7.82.1 Detailed Description	3222
7.83 locale_classes.h File Reference	3222
7.83.1 Detailed Description	3222
7.84 locale_classes.tcc File Reference	3222
7.84.1 Detailed Description	3222
7.85 locale_conv.h File Reference	3222
7.85.1 Detailed Description	3223
7.86 locale_facets.h File Reference	3223
7.86.1 Detailed Description	3225
7.87 locale_facets.tcc File Reference	3225
7.87.1 Detailed Description	3225
7.88 locale_facets_nonio.h File Reference	3225
7.88.1 Detailed Description	3226
7.89 locale_facets_nonio.tcc File Reference	3226
7.89.1 Detailed Description	3226
7.90 localefwd.h File Reference	3226
7.90.1 Detailed Description	3227
7.91 mask_array.h File Reference	3227
7.91.1 Detailed Description	3227
7.92 max_size_type.h File Reference	3227
7.92.1 Detailed Description	3227
7.93 memoryfwd.h File Reference	3227
7.93.1 Detailed Description	3227
7.94 mofunc_impl.h File Reference	3227
7.94.1 Detailed Description	3228
7.95 move.h File Reference	3228
7.95.1 Detailed Description	3228

7.96 move_only_function.h File Reference	3229
7.96.1 Detailed Description	3229
7.97 nested_exception.h File Reference	3229
7.97.1 Detailed Description	3229
7.98 new_allocator.h File Reference	3229
7.98.1 Detailed Description	3229
7.99 new_allocator.h File Reference	3229
7.99.1 Detailed Description	3230
7.100 node_handle.h File Reference	3230
7.100.1 Detailed Description	3230
7.101 ostream.tcc File Reference	3230
7.101.1 Detailed Description	3230
7.102 ostream_insert.h File Reference	3230
7.102.1 Detailed Description	3231
7.103 parse_numbers.h File Reference	3231
7.103.1 Detailed Description	3231
7.104 postypes.h File Reference	3231
7.104.1 Detailed Description	3232
7.105 predefined_ops.h File Reference	3232
7.105.1 Detailed Description	3233
7.106 ptr_traits.h File Reference	3233
7.106.1 Detailed Description	3233
7.107 quoted_string.h File Reference	3233
7.107.1 Detailed Description	3234
7.108 random.h File Reference	3234
7.108.1 Detailed Description	3238
7.109 random.tcc File Reference	3238
7.109.1 Detailed Description	3241
7.110 random.tcc File Reference	3241
7.110.1 Detailed Description	3244
7.111 range_access.h File Reference	3244
7.111.1 Detailed Description	3245
7.112 ranges_algo.h File Reference	3245
7.112.1 Detailed Description	3248
7.113 ranges_algobase.h File Reference	3248
7.113.1 Detailed Description	3249
7.114 ranges_base.h File Reference	3249
7.114.1 Detailed Description	3251
7.114.2 Variable Documentation	3251

7.115 ranges_cmp.h File Reference	3252
7.115.1 Detailed Description	3252
7.116 ranges_uninitialized.h File Reference	3252
7.116.1 Detailed Description	3252
7.117 ranges_util.h File Reference	3252
7.117.1 Detailed Description	3253
7.118 refwrap.h File Reference	3253
7.118.1 Detailed Description	3253
7.119 regex.h File Reference	3254
7.119.1 Detailed Description	3256
7.120 regex.tcc File Reference	3256
7.120.1 Detailed Description	3256
7.121 regex_automaton.h File Reference	3256
7.121.1 Detailed Description	3257
7.122 regex_automaton.tcc File Reference	3257
7.122.1 Detailed Description	3257
7.123 regex_compiler.h File Reference	3257
7.123.1 Detailed Description	3257
7.124 regex_compiler.tcc File Reference	3258
7.124.1 Detailed Description	3258
7.125 regex_constants.h File Reference	3258
7.125.1 Detailed Description	3259
7.126 regex_error.h File Reference	3259
7.126.1 Detailed Description	3260
7.127 regex_executor.h File Reference	3260
7.127.1 Detailed Description	3260
7.128 regex_executor.tcc File Reference	3260
7.128.1 Detailed Description	3260
7.129 regex_scanner.h File Reference	3261
7.129.1 Detailed Description	3261
7.130 regex_scanner.tcc File Reference	3261
7.130.1 Detailed Description	3261
7.131 semaphore_base.h File Reference	3261
7.131.1 Detailed Description	3261
7.132 shared_ptr.h File Reference	3261
7.132.1 Detailed Description	3262
7.133 shared_ptr.h File Reference	3262
7.133.1 Detailed Description	3264
7.134 shared_ptr_atomic.h File Reference	3264

7.134.1 Detailed Description	3265
7.135 shared_ptr_base.h File Reference	3265
7.135.1 Detailed Description	3266
7.136 slice_array.h File Reference	3266
7.136.1 Detailed Description	3267
7.137 specfun.h File Reference	3267
7.137.1 Detailed Description	3269
7.138 sstream.tcc File Reference	3269
7.138.1 Detailed Description	3269
7.139 std_abs.h File Reference	3269
7.139.1 Detailed Description	3269
7.140 std_function.h File Reference	3270
7.140.1 Detailed Description	3270
7.141 std_mutex.h File Reference	3270
7.141.1 Detailed Description	3271
7.142 std_thread.h File Reference	3271
7.142.1 Detailed Description	3271
7.143 stl_algo.h File Reference	3271
7.143.1 Detailed Description	3281
7.144 stl_algobase.h File Reference	3281
7.144.1 Detailed Description	3286
7.145 stl_bvector.h File Reference	3286
7.145.1 Detailed Description	3287
7.146 stl_construct.h File Reference	3287
7.146.1 Detailed Description	3287
7.147 stl_deque.h File Reference	3288
7.147.1 Detailed Description	3288
7.147.2 Macro Definition Documentation	3288
7.148 stl_function.h File Reference	3289
7.148.1 Detailed Description	3290
7.149 stl_heap.h File Reference	3290
7.149.1 Detailed Description	3292
7.150 stl_iterator.h File Reference	3292
7.150.1 Detailed Description	3296
7.151 stl_iterator.h File Reference	3296
7.151.1 Detailed Description	3297
7.152 stl_iterator_base_funcs.h File Reference	3297
7.152.1 Detailed Description	3297
7.153 stl_iterator_base_types.h File Reference	3298

7.153.1 Detailed Description	3298
7.154 stl_list.h File Reference	3298
7.154.1 Detailed Description	3299
7.155 stl_map.h File Reference	3299
7.155.1 Detailed Description	3300
7.156 stl_multimap.h File Reference	3300
7.156.1 Detailed Description	3301
7.157 stl_multiset.h File Reference	3301
7.157.1 Detailed Description	3302
7.158 stl_numeric.h File Reference	3302
7.158.1 Detailed Description	3303
7.159 stl_pair.h File Reference	3303
7.159.1 Detailed Description	3304
7.159.2 Macro Definition Documentation	3304
7.160 stl_queue.h File Reference	3304
7.160.1 Detailed Description	3305
7.161 stl_raw_storage_iter.h File Reference	3305
7.161.1 Detailed Description	3306
7.162 stl_relops.h File Reference	3306
7.162.1 Detailed Description	3306
7.163 stl_set.h File Reference	3306
7.163.1 Detailed Description	3307
7.164 stl_stack.h File Reference	3307
7.164.1 Detailed Description	3308
7.165 stl_tempbuf.h File Reference	3308
7.165.1 Detailed Description	3308
7.166 stl_tree.h File Reference	3308
7.166.1 Detailed Description	3309
7.167 stl_uninitialized.h File Reference	3309
7.167.1 Detailed Description	3309
7.168 stl_vector.h File Reference	3309
7.168.1 Detailed Description	3310
7.169 stream_iterator.h File Reference	3310
7.169.1 Detailed Description	3310
7.170 streambuf.tcc File Reference	3311
7.170.1 Detailed Description	3311
7.171 streambuf_iterator.h File Reference	3311
7.171.1 Detailed Description	3312
7.172 string_view.tcc File Reference	3312

7.172.1 Detailed Description	3312
7.173 string_view.tcc File Reference	3312
7.173.1 Detailed Description	3312
7.174 stringfwd.h File Reference	3312
7.174.1 Detailed Description	3313
7.175 this_thread_sleep.h File Reference	3313
7.175.1 Detailed Description	3313
7.176 uniform_int_dist.h File Reference	3313
7.176.1 Detailed Description	3313
7.177 unique_lock.h File Reference	3313
7.177.1 Detailed Description	3314
7.178 unique_ptr.h File Reference	3314
7.178.1 Detailed Description	3315
7.179 unordered_map.h File Reference	3315
7.179.1 Detailed Description	3317
7.180 unordered_set.h File Reference	3317
7.180.1 Detailed Description	3319
7.181 uses_allocator_args.h File Reference	3319
7.181.1 Detailed Description	3320
7.182 utility.h File Reference	3320
7.182.1 Detailed Description	3321
7.183 valarray_after.h File Reference	3321
7.183.1 Detailed Description	3331
7.184 valarray_array.h File Reference	3331
7.184.1 Detailed Description	3338
7.185 valarray_array.tcc File Reference	3339
7.185.1 Detailed Description	3339
7.186 valarray_before.h File Reference	3339
7.186.1 Detailed Description	3339
7.187 vector.tcc File Reference	3340
7.187.1 Detailed Description	3340
7.188 bitset File Reference	3340
7.188.1 Detailed Description	3341
7.189 bitset File Reference	3341
7.189.1 Detailed Description	3341
7.190 cassert File Reference	3341
7.190.1 Detailed Description	3341
7.191 ccomplex File Reference	3341
7.191.1 Detailed Description	3341

7.192 ccomplex File Reference	3342
7.192.1 Detailed Description	3342
7.193 ctype File Reference	3342
7.193.1 Detailed Description	3342
7.194 ctype File Reference	3342
7.194.1 Detailed Description	3342
7.195 cerrno File Reference	3342
7.195.1 Detailed Description	3342
7.196 cenv File Reference	3342
7.196.1 Detailed Description	3343
7.197 cenv File Reference	3343
7.197.1 Detailed Description	3343
7.198 cfloat File Reference	3343
7.198.1 Detailed Description	3343
7.199 cfloat File Reference	3343
7.199.1 Detailed Description	3343
7.200 charconv File Reference	3343
7.200.1 Detailed Description	3345
7.201 chrono File Reference	3345
7.201.1 Detailed Description	3346
7.202 chrono File Reference	3346
7.202.1 Detailed Description	3347
7.203 cinttypes File Reference	3347
7.203.1 Detailed Description	3347
7.204 cinttypes File Reference	3347
7.204.1 Detailed Description	3347
7.205 ciso646 File Reference	3347
7.205.1 Detailed Description	3347
7.206 climits File Reference	3347
7.206.1 Detailed Description	3347
7.207 climits File Reference	3348
7.207.1 Detailed Description	3348
7.208 clocale File Reference	3348
7.208.1 Detailed Description	3348
7.209 cmath File Reference	3348
7.209.1 Detailed Description	3350
7.210 cmath File Reference	3350
7.210.1 Detailed Description	3351
7.211 cmath File Reference	3351

7.211.1 Detailed Description	3353
7.212 codecv File Reference	3353
7.212.1 Detailed Description	3353
7.213 complex File Reference	3353
7.213.1 Detailed Description	3357
7.214 complex File Reference	3357
7.214.1 Detailed Description	3358
7.215 complex.h File Reference	3358
7.215.1 Detailed Description	3358
7.216 Concepts	3358
7.217 condition_variable File Reference	3360
7.217.1 Detailed Description	3361
7.218 csetjmp File Reference	3361
7.218.1 Detailed Description	3361
7.219 csignal File Reference	3361
7.219.1 Detailed Description	3361
7.220 cstdalign File Reference	3361
7.220.1 Detailed Description	3361
7.221 cstdarg File Reference	3362
7.221.1 Detailed Description	3362
7.222 cstdarg File Reference	3362
7.222.1 Detailed Description	3362
7.223 cstdbool File Reference	3362
7.223.1 Detailed Description	3362
7.224 cstdbool File Reference	3362
7.224.1 Detailed Description	3362
7.225 cstddef File Reference	3362
7.225.1 Detailed Description	3363
7.226 cstdint File Reference	3363
7.226.1 Detailed Description	3363
7.227 cstdint File Reference	3363
7.227.1 Detailed Description	3364
7.228 cstdio File Reference	3364
7.228.1 Detailed Description	3364
7.229 cstdio File Reference	3364
7.229.1 Detailed Description	3364
7.230 cstdlib File Reference	3364
7.230.1 Detailed Description	3364
7.231 cstdlib File Reference	3365

7.231.1 Detailed Description	3365
7.232 cstring File Reference	3365
7.232.1 Detailed Description	3365
7.233 ctgmath File Reference	3365
7.233.1 Detailed Description	3365
7.234 ctgmath File Reference	3365
7.234.1 Detailed Description	3365
7.235 ctime File Reference	3366
7.235.1 Detailed Description	3366
7.236 ctime File Reference	3366
7.236.1 Detailed Description	3366
7.237 cuchar File Reference	3366
7.237.1 Detailed Description	3366
7.238 cwchar File Reference	3366
7.238.1 Detailed Description	3367
7.239 wchar File Reference	3367
7.239.1 Detailed Description	3367
7.240 wctype File Reference	3367
7.240.1 Detailed Description	3367
7.241 wctype File Reference	3367
7.241.1 Detailed Description	3367
7.242 assertions.h File Reference	3368
7.242.1 Detailed Description	3368
7.243 debug.h File Reference	3368
7.243.1 Detailed Description	3368
7.244 formatter.h File Reference	3369
7.244.1 Detailed Description	3369
7.245 functions.h File Reference	3369
7.245.1 Detailed Description	3371
7.246 helper_functions.h File Reference	3371
7.246.1 Detailed Description	3373
7.247 macros.h File Reference	3373
7.247.1 Detailed Description	3374
7.247.2 Macro Definition Documentation	3374
7.248 map.h File Reference	3376
7.248.1 Detailed Description	3377
7.249 multimap.h File Reference	3377
7.249.1 Detailed Description	3378
7.250 multiset.h File Reference	3378

7.250.1 Detailed Description	3379
7.251 safe_base.h File Reference	3379
7.251.1 Detailed Description	3379
7.252 safe_container.h File Reference	3379
7.252.1 Detailed Description	3379
7.253 safe_iterator.h File Reference	3380
7.253.1 Detailed Description	3380
7.254 safe_iterator.tcc File Reference	3380
7.254.1 Detailed Description	3381
7.255 safe_local_iterator.h File Reference	3382
7.255.1 Detailed Description	3382
7.256 safe_local_iterator.tcc File Reference	3382
7.256.1 Detailed Description	3382
7.257 safe_sequence.h File Reference	3382
7.257.1 Detailed Description	3383
7.258 safe_sequence.tcc File Reference	3383
7.258.1 Detailed Description	3383
7.259 safe_unordered_base.h File Reference	3383
7.259.1 Detailed Description	3383
7.260 safe_unordered_container.h File Reference	3383
7.260.1 Detailed Description	3383
7.261 safe_unordered_container.tcc File Reference	3383
7.261.1 Detailed Description	3384
7.262 set.h File Reference	3384
7.262.1 Detailed Description	3385
7.263 decimal File Reference	3385
7.263.1 Detailed Description	3394
7.264 deque File Reference	3394
7.264.1 Detailed Description	3395
7.265 deque File Reference	3395
7.265.1 Detailed Description	3395
7.266 deque File Reference	3395
7.266.1 Detailed Description	3396
7.267 expected File Reference	3396
7.267.1 Detailed Description	3396
7.268 lfts_config.h File Reference	3396
7.268.1 Detailed Description	3396
7.269 numeric_traits.h File Reference	3396
7.269.1 Detailed Description	3396

7.270 propagate_const File Reference	3396
7.270.1 Detailed Description	3398
7.271 simd File Reference	3398
7.271.1 Detailed Description	3398
7.272 aligned_buffer.h File Reference	3398
7.272.1 Detailed Description	3398
7.273 atomicity.h File Reference	3398
7.273.1 Detailed Description	3398
7.274 bitmap_allocator.h File Reference	3399
7.274.1 Detailed Description	3399
7.274.2 Macro Definition Documentation	3399
7.275 cast.h File Reference	3399
7.275.1 Detailed Description	3400
7.276 codecvt_specializations.h File Reference	3400
7.276.1 Detailed Description	3400
7.277 concurrence.h File Reference	3400
7.277.1 Detailed Description	3401
7.278 debug_allocator.h File Reference	3401
7.278.1 Detailed Description	3401
7.279 enc_filebuf.h File Reference	3401
7.279.1 Detailed Description	3401
7.280 extptr_allocator.h File Reference	3401
7.280.1 Detailed Description	3402
7.281 malloc_allocator.h File Reference	3402
7.281.1 Detailed Description	3402
7.282 mt_allocator.h File Reference	3402
7.282.1 Detailed Description	3403
7.283 assoc_container.hpp File Reference	3403
7.283.1 Detailed Description	3403
7.284 bin_search_tree.hpp File Reference	3403
7.284.1 Detailed Description	3404
7.285 node_iterators.hpp File Reference	3404
7.285.1 Detailed Description	3404
7.286 node_iterators.hpp File Reference	3404
7.286.1 Detailed Description	3404
7.287 point_iterators.hpp File Reference	3404
7.287.1 Detailed Description	3405
7.288 r_erase_fn_imps.hpp File Reference	3405
7.288.1 Detailed Description	3405

7.289 r_erase_fn_imps.hpp File Reference	3405
7.289.1 Detailed Description	3405
7.290 rotate_fn_imps.hpp File Reference	3405
7.290.1 Detailed Description	3405
7.291 rotate_fn_imps.hpp File Reference	3405
7.291.1 Detailed Description	3405
7.292 traits.hpp File Reference	3405
7.292.1 Detailed Description	3405
7.293 traits.hpp File Reference	3405
7.293.1 Detailed Description	3406
7.294 traits.hpp File Reference	3406
7.294.1 Detailed Description	3406
7.295 traits.hpp File Reference	3406
7.295.1 Detailed Description	3406
7.296 traits.hpp File Reference	3406
7.296.1 Detailed Description	3406
7.297 traits.hpp File Reference	3407
7.297.1 Detailed Description	3407
7.298 binary_heap.hpp File Reference	3407
7.298.1 Detailed Description	3407
7.299 const_iterator.hpp File Reference	3407
7.299.1 Detailed Description	3407
7.300 const_iterator.hpp File Reference	3408
7.300.1 Detailed Description	3408
7.301 const_iterator.hpp File Reference	3408
7.301.1 Detailed Description	3408
7.302 constructors_destructor_fn_imps.hpp File Reference	3408
7.302.1 Detailed Description	3408
7.303 constructors_destructor_fn_imps.hpp File Reference	3408
7.303.1 Detailed Description	3408
7.304 constructors_destructor_fn_imps.hpp File Reference	3408
7.304.1 Detailed Description	3408
7.305 constructors_destructor_fn_imps.hpp File Reference	3408
7.305.1 Detailed Description	3408
7.306 constructors_destructor_fn_imps.hpp File Reference	3408
7.306.1 Detailed Description	3408
7.307 constructors_destructor_fn_imps.hpp File Reference	3409
7.307.1 Detailed Description	3409
7.308 constructors_destructor_fn_imps.hpp File Reference	3409

7.308.1 Detailed Description	3409
7.309 constructors_destructor_fn_imps.hpp File Reference	3409
7.309.1 Detailed Description	3409
7.310 constructors_destructor_fn_imps.hpp File Reference	3409
7.310.1 Detailed Description	3409
7.311 constructors_destructor_fn_imps.hpp File Reference	3409
7.311.1 Detailed Description	3409
7.312 constructors_destructor_fn_imps.hpp File Reference	3409
7.312.1 Detailed Description	3409
7.313 constructors_destructor_fn_imps.hpp File Reference	3409
7.313.1 Detailed Description	3409
7.314 debug_fn_imps.hpp File Reference	3409
7.314.1 Detailed Description	3409
7.315 debug_fn_imps.hpp File Reference	3409
7.315.1 Detailed Description	3409
7.316 debug_fn_imps.hpp File Reference	3409
7.316.1 Detailed Description	3409
7.317 debug_fn_imps.hpp File Reference	3410
7.317.1 Detailed Description	3410
7.318 debug_fn_imps.hpp File Reference	3410
7.318.1 Detailed Description	3410
7.319 debug_fn_imps.hpp File Reference	3410
7.319.1 Detailed Description	3410
7.320 debug_fn_imps.hpp File Reference	3410
7.320.1 Detailed Description	3410
7.321 debug_fn_imps.hpp File Reference	3410
7.321.1 Detailed Description	3410
7.322 debug_fn_imps.hpp File Reference	3410
7.322.1 Detailed Description	3410
7.323 debug_fn_imps.hpp File Reference	3410
7.323.1 Detailed Description	3410
7.324 debug_fn_imps.hpp File Reference	3410
7.324.1 Detailed Description	3410
7.325 debug_fn_imps.hpp File Reference	3410
7.325.1 Detailed Description	3410
7.326 debug_fn_imps.hpp File Reference	3410
7.326.1 Detailed Description	3410
7.327 debug_fn_imps.hpp File Reference	3411
7.327.1 Detailed Description	3411

7.328 debug_fn_imps.hpp File Reference	3411
7.328.1 Detailed Description	3411
7.329 entry_cmp.hpp File Reference	3411
7.329.1 Detailed Description	3411
7.330 entry_pred.hpp File Reference	3411
7.330.1 Detailed Description	3411
7.331 erase_fn_imps.hpp File Reference	3411
7.331.1 Detailed Description	3411
7.332 erase_fn_imps.hpp File Reference	3411
7.332.1 Detailed Description	3411
7.333 erase_fn_imps.hpp File Reference	3412
7.333.1 Detailed Description	3412
7.334 erase_fn_imps.hpp File Reference	3412
7.334.1 Detailed Description	3412
7.335 erase_fn_imps.hpp File Reference	3412
7.335.1 Detailed Description	3412
7.336 erase_fn_imps.hpp File Reference	3412
7.336.1 Detailed Description	3412
7.337 erase_fn_imps.hpp File Reference	3412
7.337.1 Detailed Description	3412
7.338 erase_fn_imps.hpp File Reference	3412
7.338.1 Detailed Description	3412
7.339 erase_fn_imps.hpp File Reference	3412
7.339.1 Detailed Description	3412
7.340 erase_fn_imps.hpp File Reference	3412
7.340.1 Detailed Description	3412
7.341 erase_fn_imps.hpp File Reference	3412
7.341.1 Detailed Description	3412
7.342 erase_fn_imps.hpp File Reference	3412
7.342.1 Detailed Description	3412
7.343 erase_fn_imps.hpp File Reference	3413
7.343.1 Detailed Description	3413
7.344 erase_fn_imps.hpp File Reference	3413
7.344.1 Detailed Description	3413
7.345 find_fn_imps.hpp File Reference	3413
7.345.1 Detailed Description	3413
7.346 find_fn_imps.hpp File Reference	3413
7.346.1 Detailed Description	3413
7.347 find_fn_imps.hpp File Reference	3413

7.347.1 Detailed Description	3413
7.348 find_fn_imps.hpp File Reference	3413
7.348.1 Detailed Description	3413
7.349 find_fn_imps.hpp File Reference	3413
7.349.1 Detailed Description	3413
7.350 find_fn_imps.hpp File Reference	3413
7.350.1 Detailed Description	3413
7.351 find_fn_imps.hpp File Reference	3413
7.351.1 Detailed Description	3413
7.352 find_fn_imps.hpp File Reference	3413
7.352.1 Detailed Description	3413
7.353 find_fn_imps.hpp File Reference	3414
7.353.1 Detailed Description	3414
7.354 find_fn_imps.hpp File Reference	3414
7.354.1 Detailed Description	3414
7.355 find_fn_imps.hpp File Reference	3414
7.355.1 Detailed Description	3414
7.356 info_fn_imps.hpp File Reference	3414
7.356.1 Detailed Description	3414
7.357 info_fn_imps.hpp File Reference	3414
7.357.1 Detailed Description	3414
7.358 info_fn_imps.hpp File Reference	3414
7.358.1 Detailed Description	3414
7.359 info_fn_imps.hpp File Reference	3414
7.359.1 Detailed Description	3414
7.360 info_fn_imps.hpp File Reference	3414
7.360.1 Detailed Description	3414
7.361 info_fn_imps.hpp File Reference	3414
7.361.1 Detailed Description	3414
7.362 info_fn_imps.hpp File Reference	3414
7.362.1 Detailed Description	3414
7.363 info_fn_imps.hpp File Reference	3415
7.363.1 Detailed Description	3415
7.364 info_fn_imps.hpp File Reference	3415
7.364.1 Detailed Description	3415
7.365 info_fn_imps.hpp File Reference	3415
7.365.1 Detailed Description	3415
7.366 insert_fn_imps.hpp File Reference	3415
7.366.1 Detailed Description	3415

7.367 insert_fn_imps.hpp File Reference	3415
7.367.1 Detailed Description	3415
7.368 insert_fn_imps.hpp File Reference	3415
7.368.1 Detailed Description	3415
7.369 insert_fn_imps.hpp File Reference	3415
7.369.1 Detailed Description	3415
7.370 insert_fn_imps.hpp File Reference	3415
7.370.1 Detailed Description	3415
7.371 insert_fn_imps.hpp File Reference	3415
7.371.1 Detailed Description	3415
7.372 insert_fn_imps.hpp File Reference	3415
7.372.1 Detailed Description	3415
7.373 insert_fn_imps.hpp File Reference	3416
7.373.1 Detailed Description	3416
7.374 insert_fn_imps.hpp File Reference	3416
7.374.1 Detailed Description	3416
7.375 insert_fn_imps.hpp File Reference	3416
7.375.1 Detailed Description	3416
7.376 insert_fn_imps.hpp File Reference	3416
7.376.1 Detailed Description	3416
7.377 insert_fn_imps.hpp File Reference	3416
7.377.1 Detailed Description	3416
7.378 insert_fn_imps.hpp File Reference	3416
7.378.1 Detailed Description	3416
7.379 iterators_fn_imps.hpp File Reference	3416
7.379.1 Detailed Description	3416
7.380 iterators_fn_imps.hpp File Reference	3416
7.380.1 Detailed Description	3416
7.381 iterators_fn_imps.hpp File Reference	3416
7.381.1 Detailed Description	3416
7.382 iterators_fn_imps.hpp File Reference	3416
7.382.1 Detailed Description	3416
7.383 iterators_fn_imps.hpp File Reference	3417
7.383.1 Detailed Description	3417
7.384 iterators_fn_imps.hpp File Reference	3417
7.384.1 Detailed Description	3417
7.385 iterators_fn_imps.hpp File Reference	3417
7.385.1 Detailed Description	3417
7.386 point_const_iterator.hpp File Reference	3417

7.386.1 Detailed Description	3417
7.387 point_const_iterator.hpp File Reference	3417
7.387.1 Detailed Description	3417
7.388 point_const_iterator.hpp File Reference	3417
7.388.1 Detailed Description	3417
7.389 policy_access_fn_imps.hpp File Reference	3418
7.389.1 Detailed Description	3418
7.390 policy_access_fn_imps.hpp File Reference	3418
7.390.1 Detailed Description	3418
7.391 policy_access_fn_imps.hpp File Reference	3418
7.391.1 Detailed Description	3418
7.392 policy_access_fn_imps.hpp File Reference	3418
7.392.1 Detailed Description	3418
7.393 policy_access_fn_imps.hpp File Reference	3418
7.393.1 Detailed Description	3418
7.394 policy_access_fn_imps.hpp File Reference	3418
7.394.1 Detailed Description	3418
7.395 policy_access_fn_imps.hpp File Reference	3418
7.395.1 Detailed Description	3418
7.396 resize_policy.hpp File Reference	3418
7.396.1 Detailed Description	3418
7.397 split_join_fn_imps.hpp File Reference	3419
7.397.1 Detailed Description	3419
7.398 split_join_fn_imps.hpp File Reference	3419
7.398.1 Detailed Description	3419
7.399 split_join_fn_imps.hpp File Reference	3419
7.399.1 Detailed Description	3419
7.400 split_join_fn_imps.hpp File Reference	3419
7.400.1 Detailed Description	3419
7.401 split_join_fn_imps.hpp File Reference	3419
7.401.1 Detailed Description	3419
7.402 split_join_fn_imps.hpp File Reference	3419
7.402.1 Detailed Description	3419
7.403 split_join_fn_imps.hpp File Reference	3419
7.403.1 Detailed Description	3419
7.404 split_join_fn_imps.hpp File Reference	3419
7.404.1 Detailed Description	3419
7.405 split_join_fn_imps.hpp File Reference	3419
7.405.1 Detailed Description	3419

7.406 trace_fn_imps.hpp File Reference	3419
7.406.1 Detailed Description	3419
7.407 trace_fn_imps.hpp File Reference	3420
7.407.1 Detailed Description	3420
7.408 trace_fn_imps.hpp File Reference	3420
7.408.1 Detailed Description	3420
7.409 trace_fn_imps.hpp File Reference	3420
7.409.1 Detailed Description	3420
7.410 trace_fn_imps.hpp File Reference	3420
7.410.1 Detailed Description	3420
7.411 trace_fn_imps.hpp File Reference	3420
7.411.1 Detailed Description	3420
7.412 trace_fn_imps.hpp File Reference	3420
7.412.1 Detailed Description	3420
7.413 trace_fn_imps.hpp File Reference	3420
7.413.1 Detailed Description	3420
7.414 binomial_heap.hpp File Reference	3420
7.414.1 Detailed Description	3421
7.415 binomial_heap_base.hpp File Reference	3421
7.415.1 Detailed Description	3421
7.416 branch_policy.hpp File Reference	3421
7.416.1 Detailed Description	3421
7.417 null_node_metadata.hpp File Reference	3421
7.417.1 Detailed Description	3421
7.418 cc_ht_map.hpp File Reference	3422
7.418.1 Detailed Description	3422
7.419 cmp_fn_imps.hpp File Reference	3422
7.419.1 Detailed Description	3422
7.420 cond_key_dtor_entry_dealtor.hpp File Reference	3422
7.420.1 Detailed Description	3422
7.421 constructor_destructor_fn_imps.hpp File Reference	3422
7.421.1 Detailed Description	3422
7.422 constructor_destructor_fn_imps.hpp File Reference	3422
7.422.1 Detailed Description	3422
7.423 constructor_destructor_fn_imps.hpp File Reference	3423
7.424 constructor_destructor_no_store_hash_fn_imps.hpp File Reference	3423
7.424.1 Detailed Description	3423
7.425 constructor_destructor_no_store_hash_fn_imps.hpp File Reference	3423
7.425.1 Detailed Description	3423

7.426 constructor_destructor_store_hash_fn_imps.hpp File Reference	3423
7.426.1 Detailed Description	3423
7.427 constructor_destructor_store_hash_fn_imps.hpp File Reference	3423
7.427.1 Detailed Description	3423
7.428 debug_no_store_hash_fn_imps.hpp File Reference	3423
7.428.1 Detailed Description	3423
7.429 debug_no_store_hash_fn_imps.hpp File Reference	3423
7.429.1 Detailed Description	3423
7.430 debug_store_hash_fn_imps.hpp File Reference	3423
7.430.1 Detailed Description	3423
7.431 debug_store_hash_fn_imps.hpp File Reference	3423
7.431.1 Detailed Description	3423
7.432 entry_list_fn_imps.hpp File Reference	3423
7.432.1 Detailed Description	3423
7.433 erase_no_store_hash_fn_imps.hpp File Reference	3424
7.433.1 Detailed Description	3424
7.434 erase_no_store_hash_fn_imps.hpp File Reference	3424
7.434.1 Detailed Description	3424
7.435 erase_store_hash_fn_imps.hpp File Reference	3424
7.435.1 Detailed Description	3424
7.436 erase_store_hash_fn_imps.hpp File Reference	3424
7.436.1 Detailed Description	3424
7.437 find_store_hash_fn_imps.hpp File Reference	3424
7.437.1 Detailed Description	3424
7.438 find_store_hash_fn_imps.hpp File Reference	3424
7.438.1 Detailed Description	3424
7.439 insert_no_store_hash_fn_imps.hpp File Reference	3424
7.439.1 Detailed Description	3424
7.440 insert_no_store_hash_fn_imps.hpp File Reference	3424
7.440.1 Detailed Description	3424
7.441 insert_store_hash_fn_imps.hpp File Reference	3424
7.441.1 Detailed Description	3424
7.442 insert_store_hash_fn_imps.hpp File Reference	3424
7.442.1 Detailed Description	3424
7.443 resize_fn_imps.hpp File Reference	3425
7.443.1 Detailed Description	3425
7.444 resize_fn_imps.hpp File Reference	3425
7.444.1 Detailed Description	3425
7.445 resize_no_store_hash_fn_imps.hpp File Reference	3425

7.445.1 Detailed Description	3425
7.446 resize_no_store_hash_fn_imps.hpp File Reference	3425
7.446.1 Detailed Description	3425
7.447 resize_store_hash_fn_imps.hpp File Reference	3425
7.447.1 Detailed Description	3425
7.448 resize_store_hash_fn_imps.hpp File Reference	3425
7.448.1 Detailed Description	3425
7.449 size_fn_imps.hpp File Reference	3425
7.449.1 Detailed Description	3425
7.450 cond_dealtor.hpp File Reference	3425
7.450.1 Detailed Description	3425
7.451 container_base_dispatch.hpp File Reference	3426
7.451.1 Detailed Description	3426
7.452 debug_map_base.hpp File Reference	3426
7.452.1 Detailed Description	3426
7.453 eq_by_less.hpp File Reference	3426
7.453.1 Detailed Description	3427
7.454 hash_eq_fn.hpp File Reference	3427
7.454.1 Detailed Description	3427
7.455 find_no_store_hash_fn_imps.hpp File Reference	3427
7.455.1 Detailed Description	3427
7.456 gp_ht_map_.hpp File Reference	3427
7.456.1 Detailed Description	3427
7.457 iterator_fn_imps.hpp File Reference	3428
7.457.1 Detailed Description	3428
7.458 direct_mask_range_hashing_imp.hpp File Reference	3428
7.458.1 Detailed Description	3428
7.459 direct_mod_range_hashing_imp.hpp File Reference	3428
7.459.1 Detailed Description	3428
7.460 linear_probe_fn_imp.hpp File Reference	3428
7.460.1 Detailed Description	3428
7.461 mask_based_range_hashing.hpp File Reference	3428
7.461.1 Detailed Description	3428
7.462 mod_based_range_hashing.hpp File Reference	3428
7.462.1 Detailed Description	3428
7.463 probe_fn_base.hpp File Reference	3428
7.463.1 Detailed Description	3429
7.464 quadratic_probe_fn_imp.hpp File Reference	3429
7.464.1 Detailed Description	3429

7.465 ranged_hash_fn.hpp File Reference	3429
7.465.1 Detailed Description	3429
7.466 ranged_probe_fn.hpp File Reference	3429
7.466.1 Detailed Description	3430
7.467 sample_probe_fn.hpp File Reference	3430
7.467.1 Detailed Description	3430
7.468 sample_range_hashing.hpp File Reference	3430
7.468.1 Detailed Description	3430
7.469 sample_ranged_hash_fn.hpp File Reference	3430
7.469.1 Detailed Description	3430
7.470 sample_ranged_probe_fn.hpp File Reference	3430
7.470.1 Detailed Description	3431
7.471 left_child_next_sibling_heap.hpp File Reference	3431
7.471.1 Detailed Description	3431
7.472 node.hpp File Reference	3431
7.472.1 Detailed Description	3431
7.473 node.hpp File Reference	3431
7.473.1 Detailed Description	3431
7.474 node.hpp File Reference	3432
7.474.1 Detailed Description	3432
7.475 entry_metadata_base.hpp File Reference	3432
7.475.1 Detailed Description	3432
7.476 lu_map.hpp File Reference	3432
7.476.1 Detailed Description	3432
7.477 lu_counter_metadata.hpp File Reference	3432
7.477.1 Detailed Description	3433
7.478 sample_update_policy.hpp File Reference	3433
7.478.1 Detailed Description	3433
7.479 ov_tree_map.hpp File Reference	3433
7.479.1 Detailed Description	3433
7.480 pairing_heap.hpp File Reference	3433
7.480.1 Detailed Description	3434
7.481 insert_join_fn_imps.hpp File Reference	3434
7.481.1 Detailed Description	3434
7.482 pat_trie.hpp File Reference	3434
7.482.1 Detailed Description	3434
7.483 pat_trie_base.hpp File Reference	3434
7.483.1 Detailed Description	3435
7.484 split_fn_imps.hpp File Reference	3435

7.484.1 Detailed Description	3435
7.485 synth_access_traits.hpp File Reference	3435
7.485.1 Detailed Description	3435
7.486 update_fn_imps.hpp File Reference	3435
7.486.1 Detailed Description	3435
7.487 priority_queue_base_dispatch.hpp File Reference	3435
7.487.1 Detailed Description	3436
7.488 rb_tree_.hpp File Reference	3436
7.488.1 Detailed Description	3436
7.489 rc.hpp File Reference	3436
7.489.1 Detailed Description	3436
7.490 rc_binomial_heap_.hpp File Reference	3436
7.490.1 Detailed Description	3437
7.491 cc_hash_max_collision_check_resize_trigger_imp.hpp File Reference	3437
7.491.1 Detailed Description	3437
7.492 hash_exponential_size_policy_imp.hpp File Reference	3437
7.492.1 Detailed Description	3437
7.493 hash_load_check_resize_trigger_imp.hpp File Reference	3437
7.493.1 Detailed Description	3437
7.494 hash_load_check_resize_trigger_size_base.hpp File Reference	3437
7.494.1 Detailed Description	3437
7.495 hash_prime_size_policy_imp.hpp File Reference	3437
7.495.1 Detailed Description	3437
7.496 hash_standard_resize_policy_imp.hpp File Reference	3437
7.496.1 Detailed Description	3437
7.497 sample_resize_policy.hpp File Reference	3438
7.497.1 Detailed Description	3438
7.498 sample_resize_trigger.hpp File Reference	3438
7.498.1 Detailed Description	3438
7.499 sample_size_policy.hpp File Reference	3438
7.499.1 Detailed Description	3438
7.500 splay_fn_imps.hpp File Reference	3438
7.500.1 Detailed Description	3438
7.501 splay_tree_.hpp File Reference	3438
7.501.1 Detailed Description	3439
7.502 standard_policies.hpp File Reference	3439
7.502.1 Detailed Description	3439
7.502.2 Enumeration Type Documentation	3439
7.503 thin_heap_.hpp File Reference	3439

7.503.1 Detailed Description	3440
7.504 node_metadata_selector.hpp File Reference	3440
7.504.1 Detailed Description	3440
7.505 node_metadata_selector.hpp File Reference	3440
7.505.1 Detailed Description	3440
7.506 order_statistics_imp.hpp File Reference	3441
7.506.1 Detailed Description	3441
7.507 order_statistics_imp.hpp File Reference	3441
7.507.1 Detailed Description	3441
7.508 sample_tree_node_update.hpp File Reference	3441
7.508.1 Detailed Description	3441
7.509 tree_trace_base.hpp File Reference	3441
7.509.1 Detailed Description	3441
7.510 prefix_search_node_update_imp.hpp File Reference	3441
7.510.1 Detailed Description	3441
7.511 sample_trie_access_traits.hpp File Reference	3441
7.511.1 Detailed Description	3441
7.512 sample_trie_node_update.hpp File Reference	3441
7.512.1 Detailed Description	3442
7.513 trie_policy_base.hpp File Reference	3442
7.513.1 Detailed Description	3442
7.514 trie_string_access_traits_imp.hpp File Reference	3442
7.514.1 Detailed Description	3442
7.515 type_utils.hpp File Reference	3442
7.515.1 Detailed Description	3442
7.516 types_traits.hpp File Reference	3443
7.516.1 Detailed Description	3443
7.517 iterator.hpp File Reference	3443
7.517.1 Detailed Description	3443
7.518 point_iterator.hpp File Reference	3443
7.518.1 Detailed Description	3443
7.519 exception.hpp File Reference	3443
7.519.1 Detailed Description	3444
7.520 hash_policy.hpp File Reference	3444
7.520.1 Detailed Description	3444
7.521 list_update_policy.hpp File Reference	3445
7.521.1 Detailed Description	3445
7.522 priority_queue.hpp File Reference	3445
7.522.1 Detailed Description	3445

7.523 tag_and_trait.hpp File Reference	3445
7.523.1 Detailed Description	3446
7.524 tree_policy.hpp File Reference	3446
7.524.1 Detailed Description	3446
7.525 trie_policy.hpp File Reference	3447
7.525.1 Detailed Description	3447
7.526 pod_char_traits.h File Reference	3447
7.526.1 Detailed Description	3447
7.527 pointer.h File Reference	3448
7.527.1 Detailed Description	3449
7.528 pool_allocator.h File Reference	3449
7.528.1 Detailed Description	3450
7.529 rb_tree File Reference	3450
7.529.1 Detailed Description	3450
7.530 rc_string_base.h File Reference	3450
7.530.1 Detailed Description	3450
7.531 rope File Reference	3450
7.531.1 Detailed Description	3453
7.532 ropeimpl.h File Reference	3453
7.532.1 Detailed Description	3454
7.533 slist File Reference	3454
7.533.1 Detailed Description	3455
7.534 sso_string_base.h File Reference	3455
7.534.1 Detailed Description	3455
7.535 stdio_filebuf.h File Reference	3455
7.535.1 Detailed Description	3455
7.536 stdio_sync_filebuf.h File Reference	3455
7.536.1 Detailed Description	3455
7.537 string_conversions.h File Reference	3456
7.537.1 Detailed Description	3456
7.538 throw_allocator.h File Reference	3456
7.538.1 Detailed Description	3457
7.539 type_traits.h File Reference	3457
7.539.1 Detailed Description	3457
7.540 typelist.h File Reference	3457
7.540.1 Detailed Description	3458
7.541 vsstring.h File Reference	3458
7.541.1 Detailed Description	3461
7.542 vsstring.tcc File Reference	3461

7.542.1 Detailed Description	3462
7.543 vstring_fwd.h File Reference	3462
7.543.1 Detailed Description	3462
7.544 vstring_util.h File Reference	3462
7.544.1 Detailed Description	3462
7.545 fenv.h File Reference	3462
7.545.1 Detailed Description	3462
7.546 filesystem File Reference	3463
7.546.1 Detailed Description	3463
7.547 filesystem File Reference	3463
7.547.1 Detailed Description	3463
7.548 forward_list File Reference	3463
7.548.1 Detailed Description	3464
7.549 forward_list File Reference	3464
7.549.1 Detailed Description	3464
7.550 forward_list File Reference	3464
7.550.1 Detailed Description	3465
7.551 fstream File Reference	3465
7.551.1 Detailed Description	3466
7.552 functional File Reference	3466
7.552.1 Detailed Description	3466
7.553 functional File Reference	3467
7.553.1 Detailed Description	3467
7.554 functional File Reference	3468
7.554.1 Detailed Description	3469
7.555 future File Reference	3470
7.555.1 Detailed Description	3471
7.556 iomanip File Reference	3471
7.556.1 Detailed Description	3472
7.557 ios File Reference	3472
7.557.1 Detailed Description	3473
7.558 iosfwd File Reference	3473
7.558.1 Detailed Description	3473
7.559 istream File Reference	3473
7.559.1 Detailed Description	3474
7.560 istream File Reference	3474
7.560.1 Detailed Description	3475
7.561 iterator File Reference	3475
7.561.1 Detailed Description	3475

7.562 iterator File Reference	3475
7.562.1 Detailed Description	3476
7.563 iterator File Reference	3476
7.563.1 Detailed Description	3476
7.564 latch File Reference	3476
7.564.1 Detailed Description	3476
7.565 limits File Reference	3476
7.565.1 Detailed Description	3478
7.566 list File Reference	3478
7.566.1 Detailed Description	3478
7.567 list File Reference	3478
7.567.1 Detailed Description	3479
7.568 list File Reference	3479
7.568.1 Detailed Description	3479
7.569 locale File Reference	3479
7.569.1 Detailed Description	3479
7.570 map File Reference	3480
7.570.1 Detailed Description	3480
7.571 map File Reference	3480
7.571.1 Detailed Description	3480
7.572 map File Reference	3480
7.572.1 Detailed Description	3481
7.573 math.h File Reference	3481
7.573.1 Detailed Description	3481
7.574 memory File Reference	3481
7.574.1 Detailed Description	3482
7.575 memory File Reference	3482
7.575.1 Detailed Description	3482
7.576 memory File Reference	3483
7.576.1 Detailed Description	3483
7.577 memory_resource File Reference	3483
7.577.1 Detailed Description	3484
7.577.2 Function Documentation	3484
7.578 memory_resource File Reference	3484
7.578.1 Detailed Description	3485
7.578.2 Function Documentation	3485
7.579 mutex File Reference	3485
7.579.1 Detailed Description	3486
7.580 numbers File Reference	3486

7.580.1 Detailed Description	3487
7.581 numeric File Reference	3487
7.581.1 Detailed Description	3487
7.582 numeric File Reference	3487
7.582.1 Detailed Description	3487
7.583 numeric File Reference	3488
7.583.1 Detailed Description	3489
7.584 numeric File Reference	3489
7.584.1 Detailed Description	3491
7.585 optional File Reference	3491
7.585.1 Detailed Description	3491
7.586 optional File Reference	3492
7.586.1 Detailed Description	3495
7.587 ostream File Reference	3495
7.587.1 Detailed Description	3496
7.588 algo.h File Reference	3496
7.588.1 Detailed Description	3506
7.589 algobase.h File Reference	3506
7.589.1 Detailed Description	3507
7.590 balanced_quicksort.h File Reference	3508
7.590.1 Detailed Description	3508
7.591 base.h File Reference	3508
7.591.1 Detailed Description	3509
7.592 basic_iterator.h File Reference	3509
7.592.1 Detailed Description	3509
7.593 checkers.h File Reference	3509
7.593.1 Detailed Description	3509
7.594 compiletime_settings.h File Reference	3509
7.594.1 Detailed Description	3510
7.594.2 Macro Definition Documentation	3510
7.595 equally_split.h File Reference	3510
7.595.1 Detailed Description	3511
7.596 features.h File Reference	3511
7.596.1 Detailed Description	3511
7.596.2 Macro Definition Documentation	3511
7.597 find.h File Reference	3513
7.597.1 Detailed Description	3513
7.598 find_selectors.h File Reference	3513
7.598.1 Detailed Description	3513

7.599 for_each.h File Reference	3514
7.599.1 Detailed Description	3514
7.600 for_each_selectors.h File Reference	3514
7.600.1 Detailed Description	3514
7.601 iterator.h File Reference	3515
7.601.1 Detailed Description	3515
7.602 list_partition.h File Reference	3515
7.602.1 Detailed Description	3515
7.603 losertree.h File Reference	3515
7.603.1 Detailed Description	3516
7.604 merge.h File Reference	3516
7.604.1 Detailed Description	3516
7.605 multiseq_selection.h File Reference	3516
7.605.1 Detailed Description	3517
7.606 multiway_merge.h File Reference	3517
7.606.1 Detailed Description	3519
7.606.2 Macro Definition Documentation	3520
7.607 multiway_mergesort.h File Reference	3520
7.607.1 Detailed Description	3520
7.608 numericfwd.h File Reference	3520
7.608.1 Detailed Description	3522
7.609 omp_loop.h File Reference	3522
7.609.1 Detailed Description	3522
7.610 omp_loop_static.h File Reference	3522
7.610.1 Detailed Description	3522
7.611 par_loop.h File Reference	3522
7.611.1 Detailed Description	3523
7.612 parallel.h File Reference	3523
7.612.1 Detailed Description	3523
7.613 partial_sum.h File Reference	3523
7.613.1 Detailed Description	3523
7.614 partition.h File Reference	3523
7.614.1 Detailed Description	3524
7.614.2 Macro Definition Documentation	3524
7.615 queue.h File Reference	3524
7.615.1 Detailed Description	3524
7.615.2 Macro Definition Documentation	3524
7.616 quicksort.h File Reference	3524
7.616.1 Detailed Description	3525

7.617 random_number.h File Reference	3525
7.617.1 Detailed Description	3525
7.618 random_shuffle.h File Reference	3525
7.618.1 Detailed Description	3526
7.619 search.h File Reference	3526
7.619.1 Detailed Description	3526
7.620 set_operations.h File Reference	3526
7.620.1 Detailed Description	3527
7.621 settings.h File Reference	3527
7.621.1 Detailed Description	3527
7.621.2 Deciding whether to run an algorithm in parallel.	3527
7.621.3 Macro Definition Documentation	3527
7.622 sort.h File Reference	3528
7.622.1 Detailed Description	3528
7.623 tags.h File Reference	3528
7.623.1 Detailed Description	3529
7.624 types.h File Reference	3529
7.624.1 Detailed Description	3529
7.625 unique_copy.h File Reference	3530
7.625.1 Detailed Description	3530
7.626 workstealing.h File Reference	3530
7.626.1 Detailed Description	3530
7.627 queue File Reference	3530
7.627.1 Detailed Description	3530
7.628 random File Reference	3531
7.628.1 Detailed Description	3531
7.629 random File Reference	3531
7.629.1 Detailed Description	3531
7.630 ranges File Reference	3531
7.630.1 Detailed Description	3534
7.631 ratio File Reference	3534
7.631.1 Detailed Description	3535
7.632 ratio File Reference	3535
7.632.1 Detailed Description	3536
7.633 ratio File Reference	3536
7.633.1 Detailed Description	3536
7.634 regex File Reference	3536
7.634.1 Detailed Description	3536
7.635 regex File Reference	3536

7.635.1 Detailed Description	3536
7.636 scoped_allocator File Reference	3537
7.636.1 Detailed Description	3537
7.637 semaphore File Reference	3537
7.637.1 Detailed Description	3537
7.638 set File Reference	3537
7.638.1 Detailed Description	3537
7.639 set File Reference	3537
7.639.1 Detailed Description	3538
7.640 set File Reference	3538
7.640.1 Detailed Description	3538
7.641 shared_mutex File Reference	3539
7.641.1 Detailed Description	3539
7.642 source_location File Reference	3539
7.642.1 Detailed Description	3539
7.643 span File Reference	3539
7.643.1 Detailed Description	3540
7.644 sstream File Reference	3540
7.644.1 Detailed Description	3541
7.645 stack File Reference	3541
7.645.1 Detailed Description	3541
7.646 stdatomic.h File Reference	3541
7.646.1 Detailed Description	3541
7.647 stdexcept File Reference	3541
7.647.1 Detailed Description	3542
7.648 stdlib.h File Reference	3542
7.648.1 Detailed Description	3542
7.649 stop_token File Reference	3542
7.649.1 Detailed Description	3542
7.650 streambuf File Reference	3542
7.650.1 Detailed Description	3543
7.651 string File Reference	3543
7.651.1 Detailed Description	3545
7.652 string File Reference	3545
7.652.1 Detailed Description	3546
7.653 string File Reference	3546
7.653.1 Detailed Description	3546
7.654 string_view File Reference	3546
7.654.1 Detailed Description	3548

7.655 string_view File Reference	3548
7.655.1 Detailed Description	3550
7.656 syncstream File Reference	3550
7.656.1 Detailed Description	3550
7.657 system_error File Reference	3550
7.657.1 Detailed Description	3550
7.658 system_error File Reference	3551
7.658.1 Detailed Description	3551
7.659 tgmth.h File Reference	3551
7.659.1 Detailed Description	3551
7.660 thread File Reference	3551
7.660.1 Detailed Description	3552
7.661 bool_set File Reference	3552
7.661.1 Detailed Description	3553
7.662 bool_set.tcc File Reference	3553
7.662.1 Detailed Description	3553
7.663 dynamic_bitset File Reference	3553
7.663.1 Detailed Description	3554
7.664 dynamic_bitset.tcc File Reference	3554
7.664.1 Detailed Description	3554
7.665 tuple File Reference	3554
7.665.1 Detailed Description	3555
7.666 tuple File Reference	3555
7.666.1 Detailed Description	3557
7.667 type_traits File Reference	3557
7.667.1 Detailed Description	3561
7.668 type_traits File Reference	3561
7.668.1 Detailed Description	3561
7.669 type_traits File Reference	3561
7.669.1 Detailed Description	3569
7.670 typeindex File Reference	3569
7.670.1 Detailed Description	3569
7.671 unordered_map File Reference	3570
7.671.1 Detailed Description	3572
7.672 unordered_map File Reference	3572
7.672.1 Detailed Description	3572
7.673 unordered_map File Reference	3572
7.673.1 Detailed Description	3573
7.674 unordered_set File Reference	3573

7.674.1 Detailed Description	3574
7.675 unordered_set File Reference	3575
7.675.1 Detailed Description	3575
7.676 unordered_set File Reference	3575
7.676.1 Detailed Description	3576
7.677 utility File Reference	3576
7.677.1 Detailed Description	3576
7.678 utility File Reference	3576
7.678.1 Detailed Description	3577
7.679 valarray File Reference	3577
7.679.1 Detailed Description	3577
7.680 variant File Reference	3577
7.680.1 Detailed Description	3580
7.681 vector File Reference	3581
7.681.1 Detailed Description	3581
7.682 vector File Reference	3581
7.682.1 Detailed Description	3582
7.683 vector File Reference	3582
7.683.1 Detailed Description	3582
7.684 atomic_word.h File Reference	3582
7.684.1 Detailed Description	3583
7.685 basic_file.h File Reference	3583
7.685.1 Detailed Description	3583
7.686 c++allocator.h File Reference	3583
7.686.1 Detailed Description	3583
7.687 c++config.h File Reference	3583
7.687.1 Detailed Description	3589
7.688 c++io.h File Reference	3589
7.688.1 Detailed Description	3590
7.689 c++locale.h File Reference	3590
7.689.1 Detailed Description	3590
7.690 c++locale_internal.h File Reference	3590
7.690.1 Detailed Description	3590
7.691 compatibility.h File Reference	3590
7.691.1 Detailed Description	3591
7.692 compatibility.h File Reference	3591
7.692.1 Detailed Description	3591
7.693 cpu_defines.h File Reference	3591
7.693.1 Detailed Description	3591

7.694 ctype_base.h File Reference	3591
7.694.1 Detailed Description	3591
7.695 ctype_inline.h File Reference	3591
7.695.1 Detailed Description	3591
7.696 cxxabi_tweaks.h File Reference	3592
7.696.1 Detailed Description	3592
7.697 error_constants.h File Reference	3592
7.697.1 Detailed Description	3592
7.698 extc++.h File Reference	3593
7.698.1 Detailed Description	3593
7.699 messages_members.h File Reference	3593
7.699.1 Detailed Description	3593
7.700 opt_random.h File Reference	3593
7.700.1 Detailed Description	3593
7.701 os_defines.h File Reference	3593
7.701.1 Detailed Description	3593
7.702 stdc++.h File Reference	3593
7.702.1 Detailed Description	3593
7.703 stdtr1c++.h File Reference	3593
7.703.1 Detailed Description	3593
7.704 time_members.h File Reference	3593
7.704.1 Detailed Description	3594
Index	3595

1 Deprecated List

Module [negators](#)

Deprecated in C++17, no longer in the standard since C++20. Use `not_fn` instead.

Module [pointer_adaptors](#)

Deprecated in C++11, no longer in the standard since C++17.

Module [ptrmem_adaptors](#)

Deprecated in C++11, no longer in the standard since C++17. Use `mem_fn` instead.

Struct [std::__is_nullptr_t< _Tp >](#)

Non-standard. Use `is_null_pointer` instead.

Struct [std::binary_function< _Arg1, _Arg2, _Result >](#)

Deprecated in C++11, no longer in the standard since C++17.

Member [std::get_unexpected\(\)](#) `noexcept`

Removed from the C++ standard in C++17

Struct `std::is_literal_type< _Tp >`

Deprecated in C++17, removed in C++20. The idea of a literal type isn't useful.

Struct `std::is_pod< _Tp >`

Deprecated in C++20. Use `is_standard_layout` && `is_trivial` instead.

Member `std::random_shuffle` (`_RandomAccessIterator __first, _RandomAccessIterator __last`)

Since C++14 `std::random_shuffle` is not part of the C++ standard. Use `std::shuffle` instead, which was introduced in C++11.

Member `std::random_shuffle` (`_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator && __rand`)

Since C++14 `std::random_shuffle` is not part of the C++ standard. Use `std::shuffle` instead, which was introduced in C++11.

Member `std::set_unexpected` (`unexpected_handler`) `noexcept`

Removed from the C++ standard in C++17

Struct `std::unary_function< _Arg, _Result >`

Deprecated in C++11, no longer in the standard since C++17.

Member `std::unexpected` ()

Removed from the C++ standard in C++17

2 Todo List

Member `__gnu_cxx::distance` (`_InputIterator __first, _InputIterator __last, _Distance & __n`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Class `__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Class `__gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Class `__gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Class `__gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Member `__gnu_cxx::power` (`_Tp __x, _Integer __n, _MonoidOperation __monoid_op`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Member `__gnu_cxx::power` (`_Tp __x, _Integer __n`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_5_style.html

Member `__gnu_cxx::random_sample` (`_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member `__gnu_cxx::random_sample` (`_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator &__rand`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member `__gnu_cxx::random_sample_n` (`_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member `__gnu_cxx::random_sample_n` (`_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator &__rand`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Struct `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Class `__gnu_cxx::rope< _CharT, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Class `__gnu_cxx::slist< _Tp, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Module `mathsf`

Provide accuracy comparisons on a per-function basis for a small number of targets.

Class `std::basic_string< _CharT, _Traits, _Alloc >`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

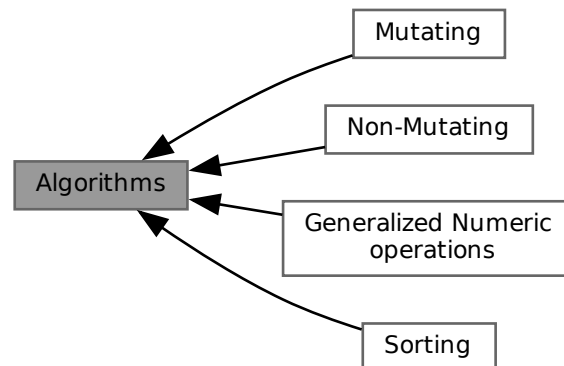
Member `std::regex_traits< _Ch_type >::transform_primary` (`_Fwd_iter __first, _Fwd_iter __last`) const

Implement this function correctly.

3 Module Documentation

3.1 Algorithms

Collaboration diagram for Algorithms:



Modules

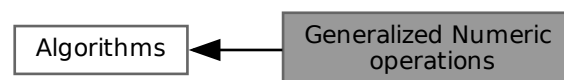
- [Generalized Numeric operations](#)
- [Mutating](#)
- [Non-Mutating](#)
- [Sorting](#)

3.1.1 Detailed Description

Components for performing algorithmic operations. Includes non-modifying sequence, modifying (mutating) sequence, sorting, searching, merge, partition, heap, set, minima, maxima, and permutation operations.

3.1.2 Generalized Numeric operations

Collaboration diagram for Generalized Numeric operations:



Functions

- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __↵
binary_op)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
__result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
__result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator std::exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
result, _Tp __init)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _OutputIterator std::exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
result, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _Tp >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator ↵
result, _BinaryOperation __binary_op, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp std::inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp ↵
__init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2`
`>`
`constexpr _Tp std::inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp ↵
__init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::iota (_ForwardIterator __first, _ForwardIterator __last, _Tp __value)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result,
__BinaryOperation __binary_op)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::value_type std::reduce (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp std::reduce (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp std::reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform_exclusive_scan (_InputIterator __first, _InputIterator __last, _Output↵
Iterator __result, _Tp __init, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _Output↵
Iterator __result, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`

- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation, typename _Tp >`
`constexpr _OutputIterator std::transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op, _UnaryOperation __unary_op, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _Tp std::transform_reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp std::transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2 >`
`constexpr _Tp std::transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`

3.1.2.1 Detailed Description

3.1.2.2 Function Documentation

accumulate() [1/2]

```
template<typename _InputIterator, typename _Tp >
constexpr _Tp std::accumulate (
    _InputIterator __first,
    _InputIterator __last,
    _Tp __init) [inline], [constexpr]
```

Accumulate values in a range.

Accumulates the values in the range `[first,last)` using `operator+()`. The initial value is *init*. The values are processed in order.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.

Returns

The final sum.

accumulate() [2/2]

```
template<typename _InputIterator, typename _Tp, typename _BinaryOperation >
constexpr _Tp std::accumulate (
    _InputIterator __first,
```

```

    _InputIterator __last,
    _Tp __init,
    _BinaryOperation __binary_op ) [inline], [constexpr]

```

Accumulate values in a range with operation.

Accumulates the values in the range `[first, last)` using the function object `__binary_op`. The initial value is `__init`. The values are processed in order.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op</code>	Function object to accumulate with.

Returns

The final sum.

adjacent_difference() [1/2]

```

template<typename _InputIterator , typename _OutputIterator >
constexpr _OutputIterator std::adjacent_difference (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result ) [constexpr]

```

Return differences between adjacent values.

Computes the difference between adjacent values in the range `[first,last)` using operator-() and writes the result to `__result`.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sums.

Returns

Iterator pointing just beyond the values written to result.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 539. `partial_sum` and `adjacent_difference` should mention requirements

adjacent_difference() [2/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation >
constexpr _OutputIterator std::adjacent_difference (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op ) [constexpr]
```

Return differences between adjacent values.

Computes the difference between adjacent values in the range [`__first`,`__last`) using the function object `__binary_op` and writes the result to `__result`.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.
<code>__binary_op</code>	Function object.

Returns

Iterator pointing just beyond the values written to result.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 539. `partial_sum` and `adjacent_difference` should mention requirements

exclusive_scan() [1/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _Tp >
constexpr _OutputIterator std::exclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _Tp __init ) [inline], [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__init</code>	Initial value.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements (and the initial value), using `std::plus<>` for summation.

This function generates an “exclusive” scan, meaning the Nth element of the output range is the sum of the first N-1 input elements, so the Nth input element is not included.

exclusive_scan() [2/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _Tp , typename _Binary↵
Operation >
constexpr _OutputIterator std::exclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _Tp __init,
    _BinaryOperation __binary_op ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__init</code>	Initial value.
<code>__binary_op</code>	Function to perform summation.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements (and the initial value), using `binary_op` for summation.

This function generates an “exclusive” scan, meaning the Nth element of the output range is the sum of the first N-1 input elements, so the Nth input element is not included.

inclusive_scan() [1/3]

```
template<typename _InputIterator , typename _OutputIterator >
constexpr _OutputIterator std::inclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result ) [inline], [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements, using `std::plus<>` for summation.

This function generates an “inclusive” scan, meaning the Nth element of the output range is the sum of the first N input elements, so the Nth input element is included.

inclusive_scan() [2/3]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation >
constexpr _OutputIterator std::inclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__binary_op</code>	Function to perform summation.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements, using `binary_op` for summation.

This function generates an “inclusive” scan, meaning the Nth element of the output range is the sum of the first N input elements, so the Nth input element is included.

inclusive_scan() [3/3]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation , typename
_Tp >
constexpr _OutputIterator std::inclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op,
    _Tp __init ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__binary_op</code>	Function to perform summation.
<code>__init</code>	Initial value.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements (and the initial value), using `binary_op` for summation.

This function generates an “inclusive” scan, meaning the Nth element of the output range is the sum of the first N input elements, so the Nth input element is included.

inner_product() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _Tp >
constexpr _Tp std::inner_product (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _Tp __init ) [inline], [constexpr]
```

Compute inner product of two ranges.

Starting with an initial value of `__init`, multiplies successive elements from the two ranges and adds each product into the accumulated value using `operator+()`. The values in the ranges are processed in order.

Parameters

<code>__first1</code>	Start of range 1.
<code>__last1</code>	End of range 1.
<code>__first2</code>	Start of range 2.
<code>__init</code>	Starting value to add other values to.

Returns

The final inner product.

inner_product() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _Tp , typename _Binary↵
Operation1 , typename _BinaryOperation2 >
constexpr _Tp std::inner_product (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _Tp __init,
    _BinaryOperation1 __binary_op1,
    _BinaryOperation2 __binary_op2 ) [inline], [constexpr]
```

Compute inner product of two ranges.

Starting with an initial value of `__init`, applies `__binary_op2` to successive elements from the two ranges and accumulates each result into the accumulated value using `__binary_op1`. The values in the ranges are processed in order.

Parameters

<code>__first1</code>	Start of range 1.
<code>__last1</code>	End of range 1.
<code>__first2</code>	Start of range 2.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op1</code>	Function object to accumulate with.
<code>__binary_op2</code>	Function object to apply to pairs of input values.

Returns

The final inner product.

iota()

```
template<typename _ForwardIterator , typename _Tp >
constexpr void std::iota (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Tp __value ) [constexpr]
```

Create a range of sequentially increasing values.

For each element in the range `[first,last)` assigns `value` and increments `value` as if by `++value`.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__value</code>	Starting value.

Returns

Nothing.

partial_sum() [1/2]

```
template<typename _InputIterator , typename _OutputIterator >
constexpr _OutputIterator std::partial_sum (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result ) [constexpr]
```

Return list of partial sums.

Accumulates the values in the range `[first,last)` using the `+` operator. As each successive input value is added into the total, that partial sum is written to `__result`. Therefore, the first value in `__result` is the first value of the input, the second value in `__result` is the sum of the first and second input values, and so on.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.

Returns

Iterator pointing just beyond the values written to `__result`.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#), and [__gnu_parallel::__sequential_random_shuffle\(\)](#).

partial_sum() [2/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation >
constexpr _OutputIterator std::partial_sum (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op ) [constexpr]
```

Return list of partial sums.

Accumulates the values in the range [first,last) using `__binary_op`. As each successive input value is added into the total, that partial sum is written to `__result`. Therefore, the first value in `__result` is the first value of the input, the second value in `__result` is the sum of the first and second input values, and so on.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.
<code>__binary_op</code>	Function object.

Returns

Iterator pointing just beyond the values written to `__result`.

reduce() [1/3]

```
template<typename _InputIterator >
constexpr iterator_traits< _InputIterator >::value_type std::reduce (
    _InputIterator __first,
    _InputIterator __last ) [inline], [constexpr]
```

Calculate reduction of values in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

The final sum.

Reduce the values in the range `[first, last)` using addition, with an initial value of `T{}`, where `T` is the iterator's value type. Equivalent to calling `std::reduce(first, last, T{}, std::plus<>())`.

reduce() [2/3]

```
template<typename _InputIterator , typename _Tp >
constexpr _Tp std::reduce (
    _InputIterator __first,
    _InputIterator __last,
    _Tp __init ) [inline], [constexpr]
```

Calculate reduction of values in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.

Returns

The final sum.

Reduce the values in the range `[first,last)` using addition. Equivalent to calling `std::reduce(first, last, init, std::plus<>())`.

reduce() [3/3]

```
template<typename _InputIterator , typename _Tp , typename _BinaryOperation >
constexpr _Tp std::reduce (
    _InputIterator __first,
    _InputIterator __last,
    _Tp __init,
    _BinaryOperation __binary_op ) [constexpr]
```

Calculate reduction of values in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op</code>	A binary function object.

Returns

The final sum.

Reduce the values in the range `[first,last)` using a binary operation. The initial value is `init`. The values are not necessarily processed in order.

This algorithm is similar to `std::accumulate` but is not required to perform the operations in order from first to last. For operations that are commutative and associative the result will be the same as for `std::accumulate`, but for other operations (such as floating point arithmetic) the result can be different.

transform_exclusive_scan()

```
template<typename _InputIterator , typename _OutputIterator , typename _Tp , typename _Binary↵
Operation , typename _UnaryOperation >
constexpr _OutputIterator std::transform_exclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _Tp __init,
    _BinaryOperation __binary_op,
    _UnaryOperation __unary_op ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__init</code>	Initial value.
<code>__binary_op</code>	Function to perform summation.
<code>__unary_op</code>	Function to transform elements of the input range.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements (and the initial value), using `__unary_op` to transform the input elements and using `__binary_op` for summation.

This function generates an “exclusive” scan, meaning the Nth element of the output range is the sum of the first N-1 input elements, so the Nth input element is not included.

transform_inclusive_scan() [1/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation , typename
_UnaryOperation >
constexpr _OutputIterator std::transform_inclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op,
    _UnaryOperation __unary_op ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__binary_op</code>	Function to perform summation.
<code>__unary_op</code>	Function to transform elements of the input range.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements, using `__unary_op` to transform the input elements and using `__binary_op` for summation.

This function generates an “inclusive” scan, meaning the Nth element of the output range is the sum of the first N input elements, so the Nth input element is included.

transform_inclusive_scan() [2/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryOperation , typename
_UnaryOperation , typename _Tp >
constexpr _OutputIterator std::transform_inclusive_scan (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryOperation __binary_op,
    _UnaryOperation __unary_op,
    _Tp __init ) [constexpr]
```

Output the cumulative sum of one range to a second range.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Start of output range.
<code>__binary_op</code>	Function to perform summation.
<code>__unary_op</code>	Function to transform elements of the input range.
<code>__init</code>	Initial value.

Returns

The end of the output range.

Write the cumulative sum (aka prefix sum, aka scan) of the input range to the output range. Each element of the output range contains the running total of all earlier elements (and the initial value), using `__unary_op` to transform the input elements and using `__binary_op` for summation.

This function generates an “inclusive” scan, meaning the Nth element of the output range is the sum of the first N input elements, so the Nth input element is included.

transform_reduce() [1/3]

```
template<typename _InputIterator , typename _Tp , typename _BinaryOperation , typename _Unary↵
Operation >
constexpr _Tp std::transform_reduce (
    _InputIterator __first,
    _InputIterator __last,
    _Tp __init,
    _BinaryOperation __binary_op,
    _UnaryOperation __unary_op ) [constexpr]
```

Transform the elements of a range and reduce.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op</code>	The function used to perform reduction.
<code>__unary_op</code>	The function used to transform values from the range.

Returns

The final sum.

Call `unary_op(first[n])` for each `n` in `[0, last-first)` and then use `binary_op` to reduce the values returned by `unary_op` to a single value of type `T`.

transform_reduce() [2/3]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _Tp >
constexpr _Tp std::transform_reduce (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _Tp __init ) [inline], [constexpr]
```

Combine elements from two ranges and reduce.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__init</code>	Starting value to add other values to.

Returns

The final sum.

Call `first1[n]*first2[n]` for each `n` in `[0, last1-first1)` and then use addition to sum those products to a single value of type `T`.

The range beginning at `first2` must contain at least `last1-first1` elements.

transform_reduce() [3/3]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _Tp , typename _Binary↵
Operation1 , typename _BinaryOperation2 >
```



```
constexpr _Tp std::transform_reduce (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _Tp __init,
    _BinaryOperation1 __binary_op1,
    _BinaryOperation2 __binary_op2 ) [constexpr]
```

Combine elements from two ranges and reduce.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op1</code>	The function used to perform reduction.
<code>__binary_op2</code>	The function used to combine values from the ranges.

Returns

The final sum.

Call `binary_op2 (first1[n], first2[n])` for each `n` in `[0, last1-first1)` and then use `binary_op1` to reduce the values returned by `binary_op2` to a single value of type `T`.

The range beginning at `first2` must contain at least `last1-first1` elements.

3.1.3 Mutating

Collaboration diagram for Mutating:



Functions

- `template<typename _II, typename _OI >`
`constexpr _OI std::copy (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`constexpr _BI2 std::copy_backward (_BI1 __first, _BI1 __last, _BI2 __result)`

- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator std::copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _↵`
`Predicate __pred)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`constexpr _OutputIterator std::copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _OI, typename _Size, typename _Tp >`
`constexpr _OI std::fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Generator >`
`constexpr void std::generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`constexpr _OutputIterator std::generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::is_partitioned (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr void std::iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`
- `template<typename _II, typename _OI >`
`constexpr _OI std::move (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`constexpr _BI2 std::move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`constexpr pair< _OutputIterator1, _OutputIterator2 > std::partition_copy (_InputIterator __first, _InputIterator ↵`
`__last, _OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate ↵`
`__pred)`
- `template<typename _RandomAccessIterator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumber↵`
`Generator &&__rand)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator std::remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result,`
`const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator ↵`
`__result, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp`
`&__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`constexpr _OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator ↵`
`__result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`constexpr void std::replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp`
`&__new_value)`

- `template<typename _BidirectionalIterator >`
`constexpr void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, _OutputIterator __result)`
- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void std::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator &&__g)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator2 std::swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred)`

3.1.3.1 Detailed Description

3.1.3.2 Function Documentation

`copy()`

```
template<typename _II, typename _OI >
constexpr _OI std::copy (
    _II __first,
    _II __last,
    _OI __result ) [inline], [constexpr]
```

Copies the range `[first,last)` into `result`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`result + (last - first)`

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling). Result may not be contained within `[first,last)`; the `copy_backward` function should be used instead.

Note that the end of the output range is permitted to be contained within `[first,last)`.

`copy_backward()`

```
template<typename _BI1 , typename _BI2 >
constexpr _BI2 std::copy_backward (
    _BI1 __first,
    _BI1 __last,
    _BI2 __result ) [inline], [constexpr]
```

Copies the range `[first,last)` into `result`.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	A bidirectional iterator.

Returns

`result - (last - first)`

The function has the same effect as `copy`, but starts at the end of the range and works its way to the start, returning the start of the result. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Result may not be in the range `(first,last]`. Use `copy` instead. Note that the start of the output range may overlap `[first,last)`.

copy_if()

```
template<typename _InputIterator , typename _OutputIterator , typename _Predicate >
constexpr _OutputIterator std::copy_if (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _Predicate __pred ) [constexpr]
```

Copy the elements of a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range [`__first`,`__last`) for which `__pred` returns true to the range beginning at `__result`.

`copy_if()` is stable, so the relative order of elements that are copied is unchanged.

copy_n()

```
template<typename _InputIterator , typename _Size , typename _OutputIterator >
constexpr _OutputIterator std::copy_n (
    _InputIterator __first,
    _Size __n,
    _OutputIterator __result ) [inline], [constexpr]
```

Copies the range [`first`,`first+n`) into [`result`,`result+n`).

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

`result+n`.

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

References [std::__iterator_category\(\)](#).

fill()

```
template<typename _ForwardIterator , typename _Tp >
constexpr void std::fill (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __value ) [inline], [constexpr]
```

Fills the range [first,last) with copies of value.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__value</code>	A reference-to-const of arbitrary type.

Returns

Nothing.

This function fills a range with copies of the same value. For char types filling contiguous areas of memory, this becomes an inline call to `memset` or `wmemset`.

fill_n()

```
template<typename _OI , typename _Size , typename _Tp >
constexpr _OI std::fill_n (
    _OI __first,
    _Size __n,
    const _Tp & __value ) [inline], [constexpr]
```

Fills the range [first,first+n) with copies of value.

Parameters

<code>__first</code>	An output iterator.
<code>__n</code>	The count of copies to perform.
<code>__value</code>	A reference-to-const of arbitrary type.

Returns

The iterator at first+n.

This function fills a range with copies of the same value. For char types filling contiguous areas of memory, this becomes an inline call to `memset` or `wmemset`.

If `__n` is negative, the function does nothing.

References [std::__iterator_category\(\)](#).

generate()

```
template<typename _ForwardIterator , typename _Generator >
constexpr void std::generate (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Generator __gen ) [constexpr]
```

Assign the result of a function object to each value in a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__gen</code>	A function object taking no arguments and returning <code>std::iterator_traits<_ForwardIterator>::value_type</code>

Returns

`generate()` returns no value.

Performs the assignment `*i = __gen ()` for each `i` in the range `[__first,__last)`.

generate_n()

```
template<typename _OutputIterator , typename _Size , typename _Generator >
constexpr _OutputIterator std::generate_n (
    _OutputIterator __first,
    _Size __n,
    _Generator __gen ) [constexpr]
```

Assign the result of a function object to each value in a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__n</code>	The length of the sequence.
<code>__gen</code>	A function object taking no arguments and returning <code>std::iterator_traits<_ForwardIterator>::value_type</code>

Returns

The end of the sequence, `__first+__n`

Performs the assignment `*i = __gen()` for each `i` in the range `[__first, __first+__n)`.

If `__n` is negative, the function does nothing and returns `__first`.

is_partitioned()

```
template<typename _InputIterator , typename _Predicate >
constexpr bool std::is_partitioned (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Checks whether the sequence is partitioned.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the range `[__first, __last)` is partitioned by `__pred`, i.e. if all elements that satisfy `__pred` appear before those that do not.

iter_swap()

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr void std::iter_swap (
    _ForwardIterator1 __a,
    _ForwardIterator2 __b ) [inline], [constexpr]
```

Swaps the contents of two iterators.

Parameters

<code>↔ __a</code>	An iterator.
<code>↔ __b</code>	Another iterator.

Returns

Nothing.

This function swaps the values pointed to by two iterators, not the iterators themselves.

References [std::swap\(\)](#).

move()

```
template<typename _II , typename _OI >
constexpr _OI std::move (
    _II __first,
    _II __last,
    _OI __result ) [inline], [constexpr]
```

Moves the range [first,last) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

result + (last - first)

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling). Result may not be contained within [first,last); the `move_backward` function should be used instead.

Note that the end of the output range is permitted to be contained within [first,last).

move_backward()

```
template<typename _BI1 , typename _BI2 >
constexpr _BI2 std::move_backward (
    _BI1 __first,
    _BI1 __last,
    _BI2 __result ) [inline], [constexpr]
```

Moves the range [first,last) into result.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	A bidirectional iterator.

Returns

result - (last - first)

The function has the same effect as `move`, but starts at the end of the range and works its way to the start, returning the start of the result. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random

access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Result may not be in the range `[first,last]`. Use `move` instead. Note that the start of the output range may overlap `[first,last]`.

partition()

```
template<typename _ForwardIterator , typename _Predicate >
constexpr _ForwardIterator std::partition (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Move elements for which a predicate is true to the beginning of a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate functor.

Returns

An iterator `middle` such that `__pred(i)` is true for each iterator `i` in the range `[__first,middle)` and false for each `i` in the range `[middle,__last)`.

`__pred` must not modify its operand. `partition()` does not preserve the relative ordering of elements in each group, use `stable_partition()` if this is needed.

References [std::__iterator_category\(\)](#), and [std::__partition\(\)](#).

partition_copy()

```
template<typename _InputIterator , typename _OutputIterator1 , typename _OutputIterator2 , typename
_Predicate >
constexpr pair< _OutputIterator1, _OutputIterator2 > std::partition_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator1 __out_true,
    _OutputIterator2 __out_false,
    _Predicate __pred ) [constexpr]
```

Copy the elements of a sequence to separate output sequences depending on the truth value of a predicate.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__out_true</code>	An output iterator.
<code>__out_false</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

A pair designating the ends of the resulting sequences.

Copies each element in the range `[__first,__last)` for which `__pred` returns true to the range beginning at `out_true` and each element for which `__pred` returns false to `__out_false`.

partition_point()

```
template<typename _ForwardIterator , typename _Predicate >
constexpr _ForwardIterator std::partition_point (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred ) [constexpr]
```

Find the partition point of a partitioned range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__pred</code>	A predicate.

Returns

An iterator `mid` such that `all_of(__first, mid, __pred)` and `none_of(mid, __last, __pred)` are both true.

References [std::advance\(\)](#), and [std::distance\(\)](#).

random_shuffle() [1/2]

```
template<typename _RandomAccessIterator >
void std::random_shuffle (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline]
```

Randomly shuffle the elements of a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

Nothing.

Reorder the elements in the range `[__first,__last)` using a random distribution, so that every possible ordering of the sequence is equally likely.

Deprecated Since C++14 `std::random_shuffle` is not part of the C++ standard. Use `std::shuffle` instead, which was introduced in C++11.

random_shuffle() [2/2]

```
template<typename _RandomAccessIterator, typename _RandomNumberGenerator >
void std::random_shuffle (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _RandomNumberGenerator && __rand )
```

Shuffle the elements of a sequence using a random number generator.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__rand</code>	The RNG functor or function.

Returns

Nothing.

Reorders the elements in the range `[__first,__last)` using `__rand` to provide a random distribution. Calling `__rand(↵N)` for a positive integer `N` should return a randomly chosen integer from the range `[0,N)`.

Deprecated Since C++14 `std::random_shuffle` is not part of the C++ standard. Use `std::shuffle` instead, which was introduced in C++11.

remove()

```
template<typename _ForwardIterator, typename _Tp >
constexpr _ForwardIterator std::remove (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __value ) [inline], [constexpr]
```

Remove elements from a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__value</code>	The value to be removed.

Returns

An iterator designating the end of the resulting sequence.

All elements equal to `__value` are removed from the range `[__first,__last)`.

`remove()` is stable, so the relative order of elements that are not removed is unchanged.

Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

remove_copy()

```
template<typename _InputIterator , typename _OutputIterator , typename _Tp >
constexpr _OutputIterator std::remove_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    const _Tp & __value ) [inline], [constexpr]
```

Copy a sequence, removing elements of a given value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__value</code>	The value to be removed.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first,__last)` not equal to `__value` to the range beginning at `__result`. `remove_copy()` is stable, so the relative order of elements that are copied is unchanged.

remove_copy_if()

```
template<typename _InputIterator , typename _OutputIterator , typename _Predicate >
constexpr _OutputIterator std::remove_copy_if (
```

```

__InputIterator __first,
__InputIterator __last,
__OutputIterator __result,
_Predicate __pred ) [inline], [constexpr]

```

Copy a sequence, removing elements for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first,__last)` for which `__pred` returns false to the range beginning at `__result`.

`remove_copy_if()` is stable, so the relative order of elements that are copied is unchanged.

`remove_if()`

```

template<typename _ForwardIterator , typename _Predicate >
constexpr _ForwardIterator std::remove_if (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred ) [inline], [constexpr]

```

Remove elements from a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

All elements for which `__pred` returns true are removed from the range `[__first,__last)`.

`remove_if()` is stable, so the relative order of elements that are not removed is unchanged.

Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

replace()

```
template<typename _ForwardIterator , typename _Tp >
constexpr void std::replace (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __old_value,
    const _Tp & __new_value ) [constexpr]
```

Replace each occurrence of one value in a sequence with another value.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__old_value</code>	The value to be replaced.
<code>__new_value</code>	The replacement value.

Returns

`replace()` returns no value.

For each iterator `i` in the range `[__first,__last)` if `*i == __old_value` then the assignment `*i = __new_value` is performed.

replace_copy_if()

```
template<typename _InputIterator , typename _OutputIterator , typename _Predicate , typename _Tp
>
constexpr _OutputIterator std::replace_copy_if (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _Predicate __pred,
    const _Tp & __new_value ) [inline], [constexpr]
```

Copy a sequence, replacing each value for which a predicate returns true with another value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.
<code>__new_value</code>	The replacement value.

Returns

The end of the output sequence, `__result+(__last-__first)`.

Copies each element in the range `[__first,__last)` to the range `[__result,__result+(__last-__first))` replacing elements for which `__pred` returns true with `__new_value`.

replace_if()

```
template<typename _ForwardIterator, typename _Predicate, typename _Tp >
constexpr void std::replace_if (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred,
    const _Tp & __new_value ) [constexpr]
```

Replace each value in a sequence for which a predicate returns true with another value.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate.
<code>__new_value</code>	The replacement value.

Returns

`replace_if()` returns no value.

For each iterator `i` in the range `[__first,__last)` if `__pred(*i)` is true then the assignment `*i = __new_value` is performed.

reverse()

```
template<typename _BidirectionalIterator >
constexpr void std::reverse (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last ) [inline], [constexpr]
```

Reverse a sequence.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.

Returns

`reverse()` returns no value.

Reverses the order of the elements in the range `[__first, __last)`, so that the first element becomes the last etc. For every `i` such that $0 \leq i < (\text{__last} - \text{__first})/2$, `reverse()` swaps `*(__first+i)` and `*(__last-(i+1))`

References [std::__iterator_category\(\)](#), and [std::__reverse\(\)](#).

reverse_copy()

```
template<typename _BidirectionalIterator, typename _OutputIterator >
constexpr _OutputIterator std::reverse_copy (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last,
    _OutputIterator __result ) [constexpr]
```

Copy a sequence, reversing its elements.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies the elements in the range `[__first, __last)` to the range `[__result, __result+(__last-__first))` such that the order of the elements is reversed. For every `i` such that $0 \leq i < (\text{__last} - \text{__first})$, `reverse_copy()` performs the assignment `*(__result+(__last-__first)-1-i) = *(__first+i)`. The ranges `[__first, __last)` and `[__result, __result+(__last-__first))` must not overlap.

rotate()

```
template<typename _ForwardIterator >
constexpr _ForwardIterator std::_V2::rotate (
    _ForwardIterator __first,
    _ForwardIterator __middle,
    _ForwardIterator __last ) [inline], [constexpr]
```

Rotate the elements of a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__middle</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

`first + (last - middle).`

Rotates the elements of the range `[__first, __last)` by `(__middle - __first)` positions so that the element at `__middle` is moved to `__first`, the element at `__middle+1` is moved to `__first+1` and so on for each element in the range `[__first, __last)`.

This effectively swaps the ranges `[__first, __middle)` and `[__middle, __last)`.

Performs `*(__first+(n+(__last-__middle))%(__last-__first))=*(__first+n)` for each `n` in the range `[0, __last-__first)`.

References [std::__iterator_category\(\)](#), and [std::__rotate\(\)](#).

rotate_copy()

```
template<typename _ForwardIterator, typename _OutputIterator>
constexpr _OutputIterator std::rotate_copy (
    _ForwardIterator __first,
    _ForwardIterator __middle,
    _ForwardIterator __last,
    _OutputIterator __result ) [inline], [constexpr]
```

Copy a sequence, rotating its elements.

Parameters

<code>__first</code>	A forward iterator.
<code>__middle</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies the elements of the range `[__first, __last)` to the range beginning at

Returns

, rotating the copied elements by `(__middle-__first)` positions so that the element at `__middle` is moved to `__result`, the element at `__middle+1` is moved to `__result+1` and so on for each element in the range `[__first, __last)`.

Performs `*(__result+(n+(__last-__middle))%(__last-__first))=*(__first+n)` for each `n` in the range `[0, __last-__first)`.

shuffle()

```
template<typename _RandomAccessIterator , typename _UniformRandomNumberGenerator >
void std::shuffle (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _UniformRandomNumberGenerator && __g )
```

Shuffle the elements of a sequence using a uniform random number generator.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__g</code>	A <code>UniformRandomNumberGenerator</code> (26.5.1.3).

Returns

Nothing.

Reorders the elements in the range `[__first,__last)` using `__g` to provide random numbers.

References [std::__gen_two_uniform_ints\(\)](#), [std::pair<_T1, _T2>::first](#), and [std::pair<_T1, _T2>::second](#).

stable_partition()

```
template<typename _ForwardIterator , typename _Predicate >
_FowardIterator std::stable_partition (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred ) [inline]
```

Move elements for which a predicate is true to the beginning of a sequence, preserving relative ordering.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate functor.

Returns

An iterator `middle` such that `__pred(i)` is true for each iterator `i` in the range `[first,middle)` and false for each `i` in the range `[middle,last)`.

Performs the same function as `partition()` with the additional guarantee that the relative ordering of elements in each group is preserved, so any two elements `x` and `y` in the range `[__first,__last)` such that `__pred(x) == __pred(y)` will have the same relative ordering after calling `stable_partition()`.

swap_ranges()

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr _ForwardIterator2 std::swap_ranges (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2 ) [constexpr]
```

Swap the elements of two sequences.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.

Returns

An iterator equal to `first2+(last1-first1)`.

Swaps each element in the range `[first1,last1)` with the corresponding element in the range `[first2,(last1-first1))`. The ranges must not overlap.

transform() [1/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _UnaryOperation >
constexpr _OutputIterator std::transform (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _UnaryOperation __unary_op ) [constexpr]
```

Perform an operation on a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__unary_op</code>	A unary operator.

Returns

An output iterator equal to `__result+(__last-__first)`.

Applies the operator to each element in the input range and assigns the results to successive elements of the output sequence. Evaluates `*(__result+N)=unary_op(*(__first+N))` for each `N` in the range `[0,__last-__first)`.

`unary_op` must not alter its argument.

transform() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename
_BinaryOperation >
constexpr _OutputIterator std::transform (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _OutputIterator __result,
    _BinaryOperation __binary_op ) [constexpr]
```

Perform an operation on corresponding elements of two sequences.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__binary_op</code>	A binary operator.

Returns

An output iterator equal to `result+(last-first)`.

Applies the operator to the corresponding elements in the two input ranges and assigns the results to successive elements of the output sequence. Evaluates `*(__result+N)=__binary_op(*(__first1+N),*(__first2+N))` for each `N` in the range `[0,__last1-__first1)`.

`binary_op` must not alter either of its arguments.

unique() [1/2]

```
template<typename _ForwardIterator >
constexpr _ForwardIterator std::unique (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Remove consecutive duplicate values from a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

An iterator designating the end of the resulting sequence.

Removes all but the first element from each group of consecutive values that compare equal. `unique()` is stable, so the relative order of elements that are not removed is unchanged. Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

unique() [2/2]

```
template<typename _ForwardIterator, typename _BinaryPredicate >
constexpr _ForwardIterator std::unique (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Remove consecutive values from a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

An iterator designating the end of the resulting sequence.

Removes all but the first element from each group of consecutive values for which `__binary_pred` returns true. `unique()` is stable, so the relative order of elements that are not removed is unchanged. Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

unique_copy() [1/2]

```
template<typename _InputIterator, typename _OutputIterator >
constexpr _OutputIterator std::unique_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result ) [inline], [constexpr]
```

Copy a sequence, removing consecutive duplicate values.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range [`__first`,`__last`) to the range beginning at `__result`, except that only the first element is copied from groups of consecutive elements that compare equal. `unique_copy()` is stable, so the relative order of elements that are copied is unchanged.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 241. Does `unique_copy()` require CopyConstructible and Assignable?

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 538. 241 again: Does `unique_copy()` require CopyConstructible and Assignable?

References [std::__iterator_category\(\)](#), and [std::__unique_copy\(\)](#).

unique_copy() [2/2]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryPredicate >
constexpr _OutputIterator std::unique_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Copy a sequence, removing consecutive values using a predicate.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range [`__first`,`__last`) to the range beginning at `__result`, except that only the first element is copied from groups of consecutive elements for which `__binary_pred` returns true. `unique_copy()` is stable, so the relative order of elements that are copied is unchanged.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 241. Does `unique_copy()` require CopyConstructible and Assignable?

References [std::__iterator_category\(\)](#), and [std::__unique_copy\(\)](#).

3.1.4 Non-Mutating

Collaboration diagram for Non-Mutating:



Functions

- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr iterator_traits< _InputIterator >::difference_type std::count (_InputIterator __first, _InputIterator __last, const _Tp & __value)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr iterator_traits< _InputIterator >::difference_type std::count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _I1, typename _I2 >`
`constexpr bool std::equal (_I1 __first1, _I1 __last1, _I2 __first2)`
- `template<typename _I1, typename _I2 >`
`constexpr bool std::equal (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _I2, typename _BinaryPredicate >`
`constexpr bool std::equal (_I1 __first1, _I1 __last1, _I2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _I1, typename _I2, typename _BinaryPredicate >`
`constexpr bool std::equal (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _InputIterator std::find (_InputIterator __first, _InputIterator __last, const _Tp & __val)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`constexpr _InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2)`

- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2,`
`_ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Function >`
`constexpr _Function std::for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _InputIterator, typename _Size, typename _Function >`
`constexpr _InputIterator std::for_each_n (_InputIterator __first, _Size __n, _Function __f)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1,`
`_InputIterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1,`
`_InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1,`
`_InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1,`
`_InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`constexpr _ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count,`
`const _Tp & __val)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count,`
`const _Tp & __val, _BinaryPredicate __binary_pred)`

3.1.4.1 Detailed Description

3.1.4.2 Function Documentation

adjacent_find() [1/2]

```
template<typename _ForwardIterator >
constexpr _ForwardIterator std::adjacent_find (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Find two adjacent values in a sequence that are equal.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

The first iterator `i` such that `i` and `i+1` are both valid iterators in `[__first,__last)` and such that `*i == *(i+1)`, or `__last` if no such iterator exists.

adjacent_find() [2/2]

```
template<typename _ForwardIterator , typename _BinaryPredicate >
constexpr _ForwardIterator std::adjacent_find (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Find two adjacent values in a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

The first iterator `i` such that `i` and `i+1` are both valid iterators in `[__first,__last)` and such that `__binary_pred(*i,*(i+1))` is true, or `__last` if no such iterator exists.

all_of()

```
template<typename _InputIterator , typename _Predicate >
constexpr bool std::all_of (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Checks that a predicate is true for all the elements of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if `__pred` is true for each element in the range `[__first,__last)`, and false otherwise.

any_of()

```
template<typename _InputIterator , typename _Predicate >
constexpr bool std::any_of (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Checks that a predicate is true for at least one element of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if an element exists in the range `[__first,__last)` such that `__pred` is true, and false otherwise.

count()

```
template<typename _InputIterator , typename _Tp >
constexpr iterator\_traits< _InputIterator >::difference_type std::count (
    _InputIterator __first,
    _InputIterator __last,
    const _Tp & __value ) [inline], [constexpr]
```

Count the number of copies of a value in a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__value</code>	The value to be counted.

Returns

The number of iterators `i` in the range `[__first,__last)` for which `*i == __value`

count_if()

```
template<typename _InputIterator , typename _Predicate >
constexpr iterator\_traits< _InputIterator >::difference_type std::count_if (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Count the elements of a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The number of iterators `i` in the range `[__first,__last)` for which `__pred(*i)` is true.

equal() [1/4]

```
template<typename _II1 , typename _II2 >
constexpr bool std::equal (
    _II1 __first1,
```

```
__II1 __last1,  
__II2 __first2 ) [inline], [constexpr]
```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.

Returns

A boolean true or false.

This compares the elements of two ranges using `==` and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

equal() [2/4]

```
template<typename _II1 , typename _II2 >  
constexpr bool std::equal (  
    __II1 __first1,  
    __II1 __last1,  
    __II2 __first2,  
    __II2 __last2 ) [inline], [constexpr]
```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A boolean true or false.

This compares the elements of two ranges using `==` and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

equal() [3/4]

```
template<typename _IIter1 , typename _IIter2 , typename _BinaryPredicate >  
constexpr bool std::equal (  

```

```

_IIter1 __first1,
_IIter1 __last1,
_IIter2 __first2,
_BinaryPredicate __binary_pred ) [inline], [constexpr]

```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A boolean true or false.

This compares the elements of two ranges using the `binary_pred` parameter, and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

`equal()` [4/4]

```

template<typename _IIter1 , typename _IIter2 , typename _BinaryPredicate >
constexpr bool std::equal (
    _IIter1 __first1,
    _IIter1 __last1,
    _IIter2 __first2,
    _IIter2 __last2,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]

```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A boolean true or false.

This compares the elements of two ranges using the `binary_pred` parameter, and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

find()

```
template<typename _InputIterator , typename _Tp >
constexpr _InputIterator std::find (
    _InputIterator __first,
    _InputIterator __last,
    const _Tp & __val ) [inline], [constexpr]
```

Find the first occurrence of a value in a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__val</code>	The value to find.

Returns

The first iterator `i` in the range `[__first,__last)` such that `*i == __val`, or `__last` if no such iterator exists.

References [std::__find_if\(\)](#).

find_end() [1/2]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr _ForwardIterator1 std::find_end (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2 ) [inline], [constexpr]
```

Find last matching subsequence in a sequence.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of sequence to match.
<code>__last2</code>	End of sequence to match.

Returns

The last iterator `i` in the range `[__first1,__last1-(__last2-__first2))` such that `*(i+N) == *(__first2+N)` for each `N` in the range `[0,__last2-__first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1,__last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2,__last2)` and returns an iterator to the `__first` element of the sub-sequence, or `__last1` if the sub-sequence is not found. The sub-sequence will be the last such subsequence contained in `[__first1,__last1)`.

Because the sub-sequence must lie completely within the range $[_first1, _last1)$ it must start at a position less than $_last1 - (_last2 - _first2)$ where $_last2 - _first2$ is the length of the sub-sequence. This means that the returned iterator i will be in the range $[_first1, _last1 - (_last2 - _first2))$

References [std::iterator_category\(\)](#).

find_end() [2/2]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 , typename _BinaryPredicate >
constexpr _ForwardIterator1 std::find_end (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2,
    _BinaryPredicate __comp ) [inline], [constexpr]
```

Find last matching subsequence in a sequence using a predicate.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of sequence to match.
<code>__last2</code>	End of sequence to match.
<code>__comp</code>	The predicate to use.

Returns

The last iterator i in the range $[_first1, _last1 - (_last2 - _first2))$ such that `predicate(*(i+N), (_first2+N))` is true for each N in the range $[0, _last2 - _first2)$, or `__last1` if no such iterator exists.

Searches the range $[_first1, _last1)$ for a sub-sequence that compares equal value-by-value with the sequence given by $[_first2, _last2)$ using `comp` as a predicate and returns an iterator to the first element of the sub-sequence, or `__last1` if the sub-sequence is not found. The sub-sequence will be the last such subsequence contained in $[_first, _last1)$.

Because the sub-sequence must lie completely within the range $[_first1, _last1)$ it must start at a position less than $_last1 - (_last2 - _first2)$ where $_last2 - _first2$ is the length of the sub-sequence. This means that the returned iterator i will be in the range $[_first1, _last1 - (_last2 - _first2))$

References [std::iterator_category\(\)](#).

find_first_of() [1/2]

```
template<typename _InputIterator , typename _ForwardIterator >
constexpr _InputIterator std::find_first_of (
    _InputIterator __first1,
    _InputIterator __last1,
    _ForwardIterator __first2,
    _ForwardIterator __last2 ) [constexpr]
```

Find element from a set in a sequence.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of match candidates.
<code>__last2</code>	End of match candidates.

Returns

The first iterator `i` in the range `[__first1, __last1)` such that `*i == *(i2)` such that `i2` is an iterator in `[__first2, __last2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for an element that is equal to some element in the range `[__first2, __last2)`. If found, returns an iterator in the range `[__first1, __last1)`, otherwise returns `__last1`.

find_first_of() [2/2]

```
template<typename _InputIterator , typename _ForwardIterator , typename _BinaryPredicate >
constexpr _InputIterator std::find_first_of (
    _InputIterator __first1,
    _InputIterator __last1,
    _ForwardIterator __first2,
    _ForwardIterator __last2,
    _BinaryPredicate __comp ) [constexpr]
```

Find element from a set in a sequence using a predicate.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of match candidates.
<code>__last2</code>	End of match candidates.
<code>__comp</code>	Predicate to use.

Returns

The first iterator `i` in the range `[__first1, __last1)` such that `comp(*i, *(i2))` is true and `i2` is an iterator in `[__first2, __last2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for an element that is equal to some element in the range `[__first2, __last2)`. If found, returns an iterator in the range `[__first1, __last1)`, otherwise returns `__last1`.

find_if()

```
template<typename _InputIterator , typename _Predicate >
constexpr _InputIterator std::find_if (
```

```

    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]

```

Find the first element in a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The first iterator `i` in the range `[__first,__last)` such that `__pred(*i)` is true, or `__last` if no such iterator exists.

References [std::__find_if\(\)](#).

find_if_not()

```

template<typename _InputIterator , typename _Predicate >
constexpr _InputIterator std::find_if_not (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]

```

Find the first element in a sequence for which a predicate is false.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The first iterator `i` in the range `[__first,__last)` such that `__pred(*i)` is false, or `__last` if no such iterator exists.

References [std::__find_if_not\(\)](#).

for_each()

```

template<typename _InputIterator , typename _Function >
constexpr _Function std::for_each (

```

```
_InputIterator __first,  
_InputIterator __last,  
_Function __f ) [constexpr]
```

Apply a function to every element of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__f</code>	A unary function object.

Returns

`__f`

Applies the function object `__f` to each element in the range `[first,last)`. `__f` must not modify the order of the sequence. If `__f` has a return value it is ignored.

for_each_n()

```
template<typename _InputIterator , typename _Size , typename _Function >  
constexpr _InputIterator std::for_each_n (  
    _InputIterator __first,  
    _Size __n,  
    _Function __f ) [constexpr]
```

Apply a function to every element of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	A value convertible to an integer.
<code>__f</code>	A unary function object.

Returns

`__first+__n`

Applies the function object `__f` to each element in the range `[first, first+n)`. `__f` must not modify the order of the sequence. If `__f` has a return value it is ignored.

References [std::move\(\)](#).

is_permutation() [1/4]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr bool std::is_permutation (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2 ) [inline], [constexpr]
```

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.

Returns

true if there exists a permutation of the elements in the range `[__first2, __first2 + (__last1 - __first1))`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, begin)` returns true; otherwise, returns false.

is_permutation() [2/4]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 , typename _BinaryPredicate >
constexpr bool std::is_permutation (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _BinaryPredicate __pred ) [inline], [constexpr]
```

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__pred</code>	A binary predicate.

Returns

true if there exists a permutation of the elements in the range `[__first2, __first2 + (__last1 - __first1))`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, __begin, __pred)` returns true; otherwise, returns false.

is_permutation() [3/4]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr bool std::is_permutation (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2 ) [inline], [constexpr]
```

Checks whether a permutaion of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of first range.

Returns

true if there exists a permutation of the elements in the range [`__first2`, `__last2`), beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, begin)` returns true; otherwise, returns false.

is_permutation() [4/4]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 , typename _BinaryPredicate >
constexpr bool std::is_permutation (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2,
    _BinaryPredicate __pred ) [inline], [constexpr]
```

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of first range.
<code>__pred</code>	A binary predicate.

Returns

true if there exists a permutation of the elements in the range [`__first2`, `__last2`), beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, __begin, __pred)` returns true; otherwise, returns false.

mismatch() [1/4]

```
template<typename _InputIterator1 , typename _InputIterator2 >
constexpr pair< _InputIterator1, _InputIterator2 > std::mismatch (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2 ) [inline], [constexpr]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using `==` and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

mismatch() [2/4]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _BinaryPredicate >
constexpr pair< _InputIterator1, _InputIterator2 > std::mismatch (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using the `binary_pred` parameter, and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

mismatch() [3/4]

```
template<typename _InputIterator1 , typename _InputIterator2 >
constexpr pair< _InputIterator1, _InputIterator2 > std::mismatch (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2 ) [inline], [constexpr]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using `==` and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

mismatch() [4/4]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _BinaryPredicate >
constexpr pair< _InputIterator1, _InputIterator2 > std::mismatch (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using the `binary_pred` parameter, and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

none_of()

```
template<typename _InputIterator , typename _Predicate >
constexpr bool std::none_of (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]
```

Checks that a predicate is false for all the elements of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if `__pred` is false for each element in the range `[__first,__last)`, and false otherwise.

search() [1/2]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 >
constexpr _ForwardIterator1 std::search (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2 ) [inline], [constexpr]
```

Search a sequence for a matching sub-sequence.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.
<code>__last2</code>	A forward iterator.

Returns

The first iterator `i` in the range `[__first1, __last1 - (__last2 - __first2))` such that `*(i+N) == *(__first2+N)` for each `N` in the range `[0, __last2 - __first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2, __last2)` and returns an iterator to the first element of the sub-sequence, or `__last1` if the sub-sequence is not found.

Because the sub-sequence must lie completely within the range `[__first1, __last1)` it must start at a position less than `__last1 - (__last2 - __first2)` where `__last2 - __first2` is the length of the sub-sequence.

This means that the returned iterator `i` will be in the range `[__first1, __last1 - (__last2 - __first2))`

search() [2/2]

```
template<typename _ForwardIterator1 , typename _ForwardIterator2 , typename _BinaryPredicate >
constexpr _ForwardIterator1 std::search (
    _ForwardIterator1 __first1,
    _ForwardIterator1 __last1,
    _ForwardIterator2 __first2,
    _ForwardIterator2 __last2,
    _BinaryPredicate __predicate ) [inline], [constexpr]
```

Search a sequence for a matching sub-sequence using a predicate.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.
<code>__last2</code>	A forward iterator.
<code>__predicate</code>	A binary predicate.

Returns

The first iterator `i` in the range `[__first1, __last1 - (__last2 - __first2))` such that `__predicate(*(i+N), *(__first2+N))` is true for each `N` in the range `[0, __last2 - __first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2, __last2)`, using `__predicate` to determine equality, and returns an iterator to the first element of the sub-sequence, or `__last1` if no such iterator exists.

See also

`search(_ForwardIter1, _ForwardIter1, _ForwardIter2, _ForwardIter2)`

search_n() [1/2]

```
template<typename _ForwardIterator, typename _Integer, typename _Tp >
constexpr _ForwardIterator std::search_n (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Integer __count,
    const _Tp & __val ) [inline], [constexpr]
```

Search a sequence for a number of consecutive values.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__count</code>	The number of consecutive values.
<code>__val</code>	The value to find.

Returns

The first iterator `i` in the range `[__first, __last - __count)` such that `*(i+N) == __val` for each `N` in the range `[0, ↵
__count)`, or `__last` if no such iterator exists.

Searches the range `[__first, __last)` for `count` consecutive elements equal to `__val`.

search_n() [2/2]

```
template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >
constexpr _ForwardIterator std::search_n (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Integer __count,
    const _Tp & __val,
    _BinaryPredicate __binary_pred ) [inline], [constexpr]
```

Search a sequence for a number of consecutive values using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__count</code>	The number of consecutive values.
<code>__val</code>	The value to find.
<code>__binary_pred</code>	A binary predicate.

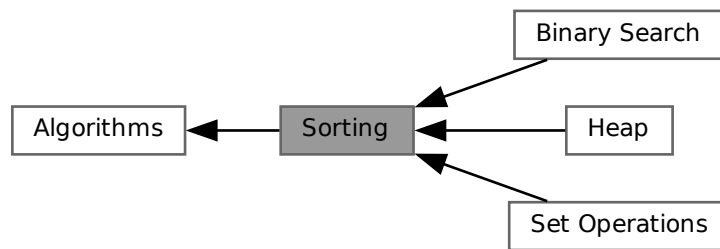
Returns

The first iterator `i` in the range `[__first, __last - __count)` such that `__binary_pred(*(i+N), __val)` is true for each `N` in the range `[0, __count)`, or `__last` if no such iterator exists.

Searches the range `[__first, __last)` for `__count` consecutive elements for which the predicate returns true.

3.1.5 Sorting

Collaboration diagram for Sorting:

**Modules**

- [Binary Search](#)
- [Heap](#)
- [Set Operations](#)

Functions

- `template<typename _Tp >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵
last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵
last, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`

- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _II1, typename _II2 >`
`constexpr bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`constexpr bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _Compare __comp)`
- `template<typename _Tp >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Tp >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`constexpr bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`constexpr bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

3.1.5.1 Detailed Description

3.1.5.2 Function Documentation

clamp() [1/2]

```
template<typename _Tp >
constexpr const _Tp & std::clamp (
    const _Tp & __val,
    const _Tp & __lo,
    const _Tp & __hi ) [constexpr]
```

Returns the value clamped between lo and hi.

Parameters

<code>__val</code>	A value of arbitrary type.
<code>__lo</code>	A lower limit of arbitrary type.
<code>__hi</code>	An upper limit of arbitrary type.

Return values

<code>`__lo`</code>	if <code>__val < __lo</code>
<code>`__hi`</code>	if <code>__hi < __val</code>
<code>`__val`</code>	otherwise.

Precondition

`__Tp` is `LessThanComparable` and `(__hi < __lo)` is false.

References [std::max\(\)](#), and [std::min\(\)](#).

clamp() [2/2]

```
template<typename _Tp , typename _Compare >
constexpr const _Tp & std::clamp (
    const _Tp & __val,
    const _Tp & __lo,
    const _Tp & __hi,
    _Compare __comp ) [constexpr]
```

Returns the value clamped between `lo` and `hi`.

Parameters

<code>__val</code>	A value of arbitrary type.
<code>__lo</code>	A lower limit of arbitrary type.
<code>__hi</code>	An upper limit of arbitrary type.
<code>__comp</code>	A comparison functor.

Return values

<code>`__lo`</code>	if <code>__comp(__val, __lo)</code>
<code>`__hi`</code>	if <code>__comp(__hi, __val)</code>
<code>`__val`</code>	otherwise.

Precondition

`__comp(__hi, __lo)` is false.

References [std::max\(\)](#), and [std::min\(\)](#).

inplace_merge() [1/2]

```
template<typename _BidirectionalIterator >
void std::inplace_merge (
```

```

_BidirectionalIterator __first,
_BidirectionalIterator __middle,
_BidirectionalIterator __last ) [inline]

```

Merges two sorted ranges in place.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Merges two sorted and consecutive ranges, [`__first`,`__middle`) and [`__middle`,`__last`), and puts the result in [`__first`,`__last`). The output will be sorted. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

If enough additional memory is available, this takes (`__last`-`__first`)-1 comparisons. Otherwise an NlogN algorithm is used, where N is `distance(__first,__last)`.

inplace_merge() [2/2]

```

template<typename _BidirectionalIterator , typename _Compare >
void std::inplace_merge (
    _BidirectionalIterator __first,
    _BidirectionalIterator __middle,
    _BidirectionalIterator __last,
    _Compare __comp ) [inline]

```

Merges two sorted ranges in place.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A functor to use for comparisons.

Returns

Nothing.

Merges two sorted and consecutive ranges, [`__first`,`__middle`) and [`__middle`,`__last`), and puts the result in [`__first`,`__last`). The output will be sorted. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

If enough additional memory is available, this takes $(_last - _first) - 1$ comparisons. Otherwise an $N \log N$ algorithm is used, where N is $\text{distance}(_first, _last)$.

The comparison function should have the same effects on ordering as the function used for the initial sort.

is_sorted() [1/2]

```
template<typename _ForwardIterator >
constexpr bool std::is_sorted (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Determines whether the elements of a sequence are sorted.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

True if the elements are sorted, false otherwise.

is_sorted() [2/2]

```
template<typename _ForwardIterator , typename _Compare >
constexpr bool std::is_sorted (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Determines whether the elements of a sequence are sorted according to a comparison functor.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

True if the elements are sorted, false otherwise.

is_sorted_until() [1/2]

```
template<typename _ForwardIterator >
```



```
constexpr _ForwardIterator std::is_sorted_until (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Determines the end of a sorted sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

An iterator pointing to the last iterator `i` in `[__first, __last)` for which the range `[__first, i)` is sorted.

`is_sorted_until()` [2/2]

```
template<typename _ForwardIterator , typename _Compare >
constexpr _ForwardIterator std::is_sorted_until (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Determines the end of a sorted sequence using comparison functor.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

An iterator pointing to the last iterator `i` in `[__first, __last)` for which the range `[__first, i)` is sorted.

`lexicographical_compare()` [1/2]

```
template<typename _II1 , typename _II2 >
constexpr bool std::lexicographical_compare (
    _II1 __first1,
    _II1 __last1,
    _II2 __first2,
    _II2 __last2 ) [inline], [constexpr]
```

Performs **dictionary** comparison on ranges.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A boolean true or false.

Returns true if the sequence of elements defined by the range `[first1,last1)` is lexicographically less than the sequence of elements defined by the range `[first2,last2)`. Returns false otherwise. (Quoted from [25.3.8]/1.) If the iterators are all character pointers, then this is an inline call to `memcmp`.

lexicographical_compare() [2/2]

```
template<typename _II1 , typename _II2 , typename _Compare >
constexpr bool std::lexicographical_compare (
    _II1 __first1,
    _II1 __last1,
    _II2 __first2,
    _II2 __last2,
    _Compare __comp ) [inline], [constexpr]
```

Performs **dictionary** comparison on ranges.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__comp</code>	A comparison functor .

Returns

A boolean true or false.

The same as the four-parameter `lexicographical_compare`, but uses the `comp` parameter instead of `<`.

max() [1/2]

```
template<typename _Tp >
constexpr const _Tp & std::max (
```

```
const _Tp & __a,  
const _Tp & __b ) [inline], [constexpr]
```

This does what you think it does.

Parameters

$_a$	A thing of arbitrary type.
$_b$	Another thing of arbitrary type.

Returns

The greater of the parameters.

This is the simple classic generic implementation. It will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Referenced by `__gnu_parallel::__parallel_nth_element()`, `std::Deque_base<_Tp, _Alloc>::_M_initialize_map()`, `std::deque<_Tp, _Alloc>::_M_reallocate_map()`, `std::clamp()`, `std::clamp()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`.

max() [2/2]

```
template<typename _Tp, typename _Compare>
constexpr const _Tp & std::max (
    const _Tp & __a,
    const _Tp & __b,
    _Compare __comp ) [inline], [constexpr]
```

This does what you think it does.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

The greater of the parameters.

This will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

max_element() [1/2]

```
template<typename _ForwardIterator>
constexpr _ForwardIterator std::max_element (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Return the maximum element in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

Iterator referencing the first instance of the largest value.

max_element() [2/2]

```
template<typename _ForwardIterator , typename _Compare >
constexpr _ForwardIterator std::max_element (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Return the maximum element in a range using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

Iterator referencing the first instance of the largest value according to `__comp`.

merge() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator >
constexpr _OutputIterator std::merge (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result ) [inline], [constexpr]
```

Merges two sorted ranges.

Parameters

<code>__first1</code>	An iterator.
<code>__first2</code>	Another iterator.
<code>__last1</code>	Another iterator.
<code>__last2</code>	Another iterator.
<code>__result</code>	An iterator pointing to the end of the merged range.

Returns

An output iterator equal to `__result + (__last1 - __first1)`

- `(__last2 - __first2)`.

Merges the ranges `[__first1, __last1)` and `[__first2, __last2)` into the sorted range `[__result, __result + (__last1 - __first1) + (__last2 - __first2))`. Both input ranges must be sorted, and the output range must not overlap with either of the input ranges. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

merge() [2/2]

```
template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename
_Compare >
constexpr _OutputIterator std::merge (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp ) [inline], [constexpr]
```

Merges two sorted ranges.

Parameters

<code>__first1</code>	An iterator.
<code>__first2</code>	Another iterator.
<code>__last1</code>	Another iterator.
<code>__last2</code>	Another iterator.
<code>__result</code>	An iterator pointing to the end of the merged range.
<code>__comp</code>	A functor to use for comparisons.

Returns

An output iterator equal to `__result + (__last1 - __first1)`

- `(__last2 - __first2)`.

Merges the ranges `[__first1, __last1)` and `[__first2, __last2)` into the sorted range `[__result, __result + (__last1 - __first1) + (__last2 - __first2))`. Both input ranges must be sorted, and the output range must not overlap with either of the input ranges. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.



The comparison function should have the same effects on ordering as the function used for the initial sort.

min() [1/2]

```
template<typename _Tp >
constexpr const _Tp & std::min (
    const _Tp & __a,
    const _Tp & __b ) [inline], [constexpr]
```

This does what you think it does.

Parameters

 <code>__a</code>	A thing of arbitrary type.
 <code>__b</code>	Another thing of arbitrary type.

Returns

The lesser of the parameters.

This is the simple classic generic implementation. It will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), [__gnu_parallel::__parallel_sort_qs_divide\(\)](#), [std::__sample\(\)](#), [__gnu_parallel::__search_template\(\)](#), [__gnu_parallel::__sequential_random_shuffle\(\)](#), [std::clamp\(\)](#), [std::clamp\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::generate_canonical\(\)](#), [__gnu_parallel::multiseq_partition\(\)](#), [__gnu_parallel::multiseq_selection\(\)](#), [std::shuffle_order_engine<_RandomNumberEngine, __k>::operator\(\)\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::seekpos\(\)](#), [std::basic_streambuf<_CharT, _Traits>::xsgetn\(\)](#), [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::xsputn\(\)](#).

min() [2/2]

```
template<typename _Tp , typename _Compare >
constexpr const _Tp & std::min (
    const _Tp & __a,
    const _Tp & __b,
    _Compare __comp ) [inline], [constexpr]
```

This does what you think it does.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

The lesser of the parameters.

This will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

min_element() [1/2]

```
template<typename _ForwardIterator >
constexpr _ForwardIterator std::min_element (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Return the minimum element in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

Iterator referencing the first instance of the smallest value.

min_element() [2/2]

```
template<typename _ForwardIterator , typename _Compare >
constexpr _ForwardIterator std::min_element (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Return the minimum element in a range using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

Iterator referencing the first instance of the smallest value according to `__comp`.

minmax() [1/2]

```
template<typename _Tp >
constexpr pair< const _Tp &, const _Tp & > std::minmax (
    const _Tp & __a,
    const _Tp & __b ) [inline], [constexpr]
```

Determines min and max at once as an ordered pair.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.

Returns

A pair(`__b`, `__a`) if `__b` is smaller than `__a`, pair(`__a`, `__b`) otherwise.

minmax() [2/2]

```
template<typename _Tp , typename _Compare >
constexpr pair< const _Tp &, const _Tp & > std::minmax (
    const _Tp & __a,
    const _Tp & __b,
    _Compare __comp ) [inline], [constexpr]
```

Determines min and max at once as an ordered pair.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

A pair(`__b`, `__a`) if `__b` is smaller than `__a`, pair(`__a`, `__b`) otherwise.

minmax_element() [1/2]

```
template<typename _ForwardIterator >
constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Return a pair of iterators pointing to the minimum and maximum elements in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

make_pair(m, M), where m is the first iterator i in [`__first`, `__last`) such that no other element in the range is smaller, and where M is the last iterator i in [`__first`, `__last`) such that no other element in the range is larger.

minmax_element() [2/2]

```
template<typename _ForwardIterator , typename _Compare >
constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Return a pair of iterators pointing to the minimum and maximum elements in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

make_pair(m, M), where m is the first iterator i in [`__first`, `__last`) such that no other element in the range is smaller, and where M is the last iterator i in [`__first`, `__last`) such that no other element in the range is larger.

next_permutation() [1/2]

```
template<typename _BidirectionalIterator >
constexpr bool std::next_permutation (
```

```
_BidirectionalIterator __first,  
_BidirectionalIterator __last ) [inline], [constexpr]
```

Permute range into the next *dictionary* ordering.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

False if wrapped to first permutation, true otherwise.

Treats all permutations of the range as a set of *dictionary* sorted sequences. Permutes the current sequence into the next one of this set. Returns true if there are more sequences to generate. If the sequence is the largest of the set, the smallest is generated and false returned.

next_permutation() [2/2]

```
template<typename _BidirectionalIterator , typename _Compare >
constexpr bool std::next_permutation (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Permute range into the next *dictionary* ordering using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	A comparison functor.

Returns

False if wrapped to first permutation, true otherwise.

Treats all permutations of the range [`__first`,`__last`) as a set of *dictionary* sorted sequences ordered by `__comp`. Permutes the current sequence into the next one of this set. Returns true if there are more sequences to generate. If the sequence is the largest of the set, the smallest is generated and false returned.

nth_element() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::nth_element (
    _RandomAccessIterator __first,
    _RandomAccessIterator __nth,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Sort a sequence just enough to find a particular position.

Parameters

<code>__first</code>	An iterator.
<code>__nth</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Rearranges the elements in the range `[__first,__last)` so that `*__nth` is the same element that would have been in that position had the whole sequence been sorted. The elements either side of `*__nth` are not completely sorted, but for any iterator *i* in the range `[__first,__nth)` and any iterator *j* in the range `[__nth,__last)` it holds that `*j < *i` is false.

References [std::__lg\(\)](#).

nth_element() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::nth_element (
    _RandomAccessIterator __first,
    _RandomAccessIterator __nth,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Sort a sequence just enough to find a particular position using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__nth</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Rearranges the elements in the range `[__first,__last)` so that `*__nth` is the same element that would have been in that position had the whole sequence been sorted. The elements either side of `*__nth` are not completely sorted, but for any iterator *i* in the range `[__first,__nth)` and any iterator *j* in the range `[__nth,__last)` it holds that `__comp(*j,*i)` is false.

References [std::__lg\(\)](#).

partial_sort() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::partial_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __middle,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Sort the smallest elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the smallest (`__middle-__first`) elements in the range `[first,last)` and moves them to the range `[__first,__middle)`. The order of the remaining elements in the range `[__middle,__last)` is undefined. After the sort if *i* and *j* are iterators in the range `[__first,__middle)` such that *i* precedes *j* and *k* is an iterator in the range `[__middle,__last)` then `*j < *i` and `*k < *i` are both false.

partial_sort() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::partial_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __middle,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Sort the smallest elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Sorts the smallest (`__middle-__first`) elements in the range `[__first,__last)` and moves them to the range `[__first,__middle)`. The order of the remaining elements in the range `[__middle,__last)` is undefined. After the sort if *i* and *j* are iterators in the range `[__first,__middle)` such that *i* precedes *j* and *k* is an iterator in the range `[__middle,__last)` then `*__comp(j,*i)` and `*__comp(*k,*i)` are both false.

partial_sort_copy() [1/2]

```
template<typename _InputIterator , typename _RandomAccessIterator >
constexpr _RandomAccessIterator std::partial_sort_copy (
    _InputIterator __first,
    _InputIterator __last,
    _RandomAccessIterator __result_first,
    _RandomAccessIterator __result_last ) [inline], [constexpr]
```

Copy the smallest elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__result_first</code>	A random-access iterator.
<code>__result_last</code>	Another random-access iterator.

Returns

An iterator indicating the end of the resulting sequence.

Copies and sorts the smallest *N* values from the range `[__first,__last)` to the range beginning at `__result_first`, where the number of elements to be copied, *N*, is the smaller of `(__last-__first)` and `(__result_last-__result_first)`. After the sort if *i* and *j* are iterators in the range `[__result_first,__result_first+N)` such that *i* precedes *j* then `*j<*i` is false. The value returned is `__result_first+N`.

partial_sort_copy() [2/2]

```
template<typename _InputIterator , typename _RandomAccessIterator , typename _Compare >
constexpr _RandomAccessIterator std::partial_sort_copy (
    _InputIterator __first,
    _InputIterator __last,
    _RandomAccessIterator __result_first,
    _RandomAccessIterator __result_last,
    _Compare __comp ) [inline], [constexpr]
```

Copy the smallest elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	Another input iterator.
<code>__result_first</code>	A random-access iterator.
<code>__result_last</code>	Another random-access iterator.
<code>__comp</code>	A comparison functor.

Returns

An iterator indicating the end of the resulting sequence.

Copies and sorts the smallest N values from the range $[_\text{first}, _\text{last})$ to the range beginning at `result_first`, where the number of elements to be copied, N , is the smaller of $(_\text{last} - _\text{first})$ and $(_\text{result_last} - _\text{result_first})$. After the sort if i and j are iterators in the range $[_\text{result_first}, _\text{result_first} + N)$ such that i precedes j then `__comp(*j, *i)` is false. The value returned is `__result_first + N`.

prev_permutation() [1/2]

```
template<typename _BidirectionalIterator >
constexpr bool std::prev_permutation (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last ) [inline], [constexpr]
```

Permute range into the previous *dictionary* ordering.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

False if wrapped to last permutation, true otherwise.

Treats all permutations of the range as a set of *dictionary* sorted sequences. Permutes the current sequence into the previous one of this set. Returns true if there are more sequences to generate. If the sequence is the smallest of the set, the largest is generated and false returned.

prev_permutation() [2/2]

```
template<typename _BidirectionalIterator , typename _Compare >
constexpr bool std::prev_permutation (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Permute range into the previous *dictionary* ordering using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	A comparison functor.

Returns

False if wrapped to last permutation, true otherwise.

Treats all permutations of the range `[__first,__last)` as a set of *dictionary* sorted sequences ordered by `__comp`. Permutes the current sequence into the previous one of this set. Returns true if there are more sequences to generate. If the sequence is the smallest of the set, the largest is generated and false returned.

sort() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Sort the elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the elements in the range `[__first,__last)` in ascending order, such that for each iterator *i* in the range `[__first,__last-1)`, `*i < *(i+1)` is false.

The relative ordering of equivalent elements is not preserved, use `stable_sort()` if this is needed.

sort() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Sort the elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Sorts the elements in the range `[__first,__last)` in ascending order, such that `__comp(*(i+1),*i)` is false for every iterator `i` in the range `[__first,__last-1)`.

The relative ordering of equivalent elements is not preserved, use `stable_sort()` if this is needed.

stable_sort() [1/2]

```
template<typename _RandomAccessIterator >
void std::stable_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline]
```

Sort the elements of a sequence, preserving the relative order of equivalent elements.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the elements in the range `[__first,__last)` in ascending order, such that for each iterator `i` in the range `[__first,__last-1)`, `*(i+1) < *i` is false.

The relative ordering of equivalent elements is preserved, so any two elements `x` and `y` in the range `[__first,__last)` such that `x < y` is false and `y < x` is false will have the same relative ordering after calling `stable_sort()`.

stable_sort() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
void std::stable_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline]
```

Sort the elements of a sequence using a predicate for comparison, preserving the relative order of equivalent elements.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Sorts the elements in the range `[__first, __last)` in ascending order, such that for each iterator `i` in the range `[__first, __last-1)`, `__comp(*(i+1), *i)` is false.

The relative ordering of equivalent elements is preserved, so any two elements `x` and `y` in the range `[__first, __last)` such that `__comp(x, y)` is false and `__comp(y, x)` is false will have the same relative ordering after calling `stable_sort()`.

3.1.5.3 Binary Search

Collaboration diagram for Binary Search:



Functions

- `template<typename _ForwardIterator, typename _Tp >`
`constexpr bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`

3.1.5.3.1 Detailed Description

These algorithms are variations of a classic binary search, and all assume that the sequence being searched is already sorted.

The number of comparisons will be logarithmic (and as few as possible). The number of steps through the sequence will be logarithmic for random-access iterators (e.g., pointers), and linear otherwise.

The LWG has passed Defect Report 270, which notes: *The proposed resolution reinterprets binary search. Instead of thinking about searching for a value in a sorted range, we view that as an important special case of a more general algorithm: searching for the partition point in a partitioned range. We also add a guarantee that the old wording did not: we ensure that the upper bound is no earlier than the lower bound, that the pair returned by equal_range is a valid range, and that the first part of that pair is the lower bound.*

The actual effect of the first sentence is that a comparison functor passed by the user doesn't necessarily need to induce a strict weak ordering relation. Rather, it partitions the range.

3.1.5.3.2 Function Documentation

binary_search() [1/2]

```
template<typename _ForwardIterator, typename _Tp>
constexpr bool std::binary_search (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val ) [constexpr]
```

Determines whether an element exists in a range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

True if `__val` (or its equivalent) is in `[__first,__last]`.

Note that this does not actually return an iterator to `__val`. For that, use `std::find` or a container's specialized `find` member functions.

binary_search() [2/2]

```
template<typename _ForwardIterator, typename _Tp, typename _Compare>
constexpr bool std::binary_search (
    _ForwardIterator __first,
```

```

_FowardIterator __last,
const _Tp & __val,
_Compare __comp ) [constexpr]

```

Determines whether an element exists in a range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

True if `__val` (or its equivalent) is in `[__first,__last]`.

Note that this does not actually return an iterator to `__val`. For that, use `std::find` or a container's specialized `find` member functions.

The comparison function should have the same effects on ordering as the function used for the initial sort.

equal_range() [1/2]

```

template<typename _ForwardIterator , typename _Tp >
constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val ) [inline], [constexpr]

```

Finds the largest subrange in which `__val` could be inserted at any place in it without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An pair of iterators defining the subrange.

This is equivalent to

```

std::make_pair(lower_bound(__first, __last, __val),
               upper_bound(__first, __last, __val))

```

but does not actually call those functions.

equal_range() [2/2]

```
template<typename _ForwardIterator, typename _Tp, typename _Compare>
constexpr pair<_ForwardIterator, _ForwardIterator> std::equal_range (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val,
    _Compare __comp) [inline], [constexpr]
```

Finds the largest subrange in which `__val` could be inserted at any place in it without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An pair of iterators defining the subrange.

This is equivalent to

```
std::make_pair(lower_bound(__first, __last, __val, __comp),
               upper_bound(__first, __last, __val, __comp))
```

but does not actually call those functions.

lower_bound() [1/2]

```
template<typename _ForwardIterator, typename _Tp>
constexpr _ForwardIterator std::lower_bound (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val) [inline], [constexpr]
```

Finds the first position in which `val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An iterator pointing to the first element *not less than val*, or `end()` if every element is less than *val*.

lower_bound() [2/2]

```
template<typename _ForwardIterator , typename _Tp , typename _Compare >
constexpr _ForwardIterator std::lower_bound (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val,
    _Compare __comp ) [inline], [constexpr]
```

Finds the first position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An iterator pointing to the first element *not less than* `__val`, or `end()` if every element is less than `__val`.

The comparison function should have the same effects on ordering as the function used for the initial sort.

upper_bound() [1/2]

```
template<typename _ForwardIterator , typename _Tp >
constexpr _ForwardIterator std::upper_bound (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val ) [inline], [constexpr]
```

Finds the last position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An iterator pointing to the first element greater than `__val`, or `end()` if no elements are greater than `__val`.

upper_bound() [2/2]

```
template<typename _ForwardIterator , typename _Tp , typename _Compare >
```

```
constexpr _ForwardIterator std::upper_bound (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __val,
    _Compare __comp ) [inline], [constexpr]
```

Finds the last position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An iterator pointing to the first element greater than `__val`, or `end()` if no elements are greater than `__val`.

The comparison function should have the same effects on ordering as the function used for the initial sort.

3.1.5.4 Heap

Collaboration diagram for Heap:



Functions

- `template<typename _RandomAccessIterator >`
`constexpr bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

- `template<typename _RandomAccessIterator >`
`constexpr void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵
__comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵
comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵
comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵
comp)`

3.1.5.4.1 Detailed Description

3.1.5.4.2 Function Documentation

is_heap() [1/2]

```
template<typename _RandomAccessIterator >
constexpr bool std::is_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Determines whether a range is a heap.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

True if range is a heap, false otherwise.

is_heap() [2/2]

```
template<typename _RandomAccessIterator, typename _Compare >
constexpr bool std::is_heap (
```

```

_RandomAccessIterator __first,
_RandomAccessIterator __last,
_Compare __comp ) [inline], [constexpr]

```

Determines whether a range is a heap using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor to use.

Returns

True if range is a heap, false otherwise.

References [std::distance\(\)](#).

is_heap_until() [1/2]

```

template<typename _RandomAccessIterator >
constexpr _RandomAccessIterator std::is_heap_until (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]

```

Search the end of a heap.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

An iterator pointing to the first element not in the heap.

This operation returns the last iterator *i* in [`__first`, `__last`) for which the range [`__first`, *i*) is a heap.

References [std::distance\(\)](#).

is_heap_until() [2/2]

```

template<typename _RandomAccessIterator , typename _Compare >
constexpr _RandomAccessIterator std::is_heap_until (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]

```

Search the end of a heap using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor to use.

Returns

An iterator pointing to the first element not in the heap.

This operation returns the last iterator `i` in `[__first, __last)` for which the range `[__first, i)` is a heap. Comparisons are made using `__comp`.

References [std::distance\(\)](#).

make_heap() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::make_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Construct a heap over a range.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.

This operation makes the elements in `[__first, __last)` into a heap.

make_heap() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::make_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Construct a heap over a range using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation makes the elements in `[__first,__last)` into a heap. Comparisons are made using `__comp`.

pop_heap() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::pop_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]
```

Pop an element off a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.

Precondition

`[__first, __last)` is a valid, non-empty range.

This operation pops the top of the heap. The elements `__first` and `__last-1` are swapped and `[__first,__last-1)` is made into a heap.

pop_heap() [2/2]

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::pop_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

Pop an element off a heap using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation pops the top of the heap. The elements `__first` and `__last-1` are swapped and `[__first,__last-1)` is made into a heap. Comparisons are made using `comp`.

push_heap() [1/2]

```
template<typename _RandomAccessIterator >
constexpr void std::push_heap (
```

```

_RandomAccessIterator __first,
_RandomAccessIterator __last ) [inline], [constexpr]

```

Push an element onto a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap + element.

This operation pushes the element at `last-1` onto the valid heap over the range `[__first,__last-1)`. After completion, `[__first,__last)` is a valid heap.

push_heap() [2/2]

```

template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::push_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]

```

Push an element onto a heap using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap + element.
<code>__comp</code>	Comparison functor.

This operation pushes the element at `__last-1` onto the valid heap over the range `[__first,__last-1)`. After completion, `[__first,__last)` is a valid heap. Compare operations are performed using `comp`.

sort_heap() [1/2]

```

template<typename _RandomAccessIterator >
constexpr void std::sort_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline], [constexpr]

```

Sort a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.

This operation sorts the valid heap in the range [`__first`,`__last`).

`sort_heap()` [2/2]

```
template<typename _RandomAccessIterator, typename _Compare>
constexpr void std::sort_heap (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp) [inline], [constexpr]
```

Sort a heap using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation sorts the valid heap in the range [`__first`,`__last`). Comparisons are made using `__comp`.

3.1.5.5 Set Operations

Collaboration diagram for Set Operations:



Functions

- `template<typename _InputIterator1, typename _InputIterator2>`
`constexpr bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare>`
`constexpr bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator>`
`constexpr _OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare>`
`constexpr _OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`

3.1.5.5.1 Detailed Description

These algorithms are common set operations performed on sequences that are already sorted. The number of comparisons will be linear.

3.1.5.5.2 Function Documentation

includes() [1/2]

```
template<typename _InputIterator1, typename _InputIterator2 >
constexpr bool std::includes (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2) [inline], [constexpr]
```

Determines whether all elements of a sequence exists in a range.

Parameters

<code>__first1</code>	Start of search range.
<code>__last1</code>	End of search range.
<code>__first2</code>	Start of sequence
<code>__last2</code>	End of sequence.

Returns

True if each element in [`__first2`,`__last2`) is contained in order within [`__first1`,`__last1`). False otherwise.

This operation expects both `[__first1, __last1)` and `[__first2, __last2)` to be sorted. Searches for the presence of each element in `[__first2, __last2)` within `[__first1, __last1)`. The iterators over each range only move forward, so this is a linear algorithm. If an element in `[__first2, __last2)` is not found before the search iterator reaches `__last2`, false is returned.

includes() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _Compare >
constexpr bool std::includes (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _Compare __comp ) [inline], [constexpr]
```

Determines whether all elements of a sequence exists in a range using comparison.

Parameters

<code>__first1</code>	Start of search range.
<code>__last1</code>	End of search range.
<code>__first2</code>	Start of sequence
<code>__last2</code>	End of sequence.
<code>__comp</code>	Comparison function to use.

Returns

True if each element in `[__first2, __last2)` is contained in order within `[__first1, __last1)` according to `comp`. False otherwise.

This operation expects both `[__first1, __last1)` and `[__first2, __last2)` to be sorted. Searches for the presence of each element in `[__first2, __last2)` within `[__first1, __last1)`, using `comp` to decide. The iterators over each range only move forward, so this is a linear algorithm. If an element in `[__first2, __last2)` is not found before the search iterator reaches `__last2`, false is returned.

set_difference() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator >
constexpr _OutputIterator std::set_difference (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result ) [inline], [constexpr]
```

Return the difference of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in the first range but not the second in order to the output range. Iterators increment for each range. When the current element of the first range is less than the second, that element is copied and the iterator advances. If the current element of the second range is less, the iterator advances, but no element is copied. If an element is contained in both ranges, no elements are copied and both ranges advance. The output range may not overlap either input range.

set_difference() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename
_Compare >
constexpr _OutputIterator std::set_difference (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp ) [inline], [constexpr]
```

Return the difference of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in the first range but not the second in order to the output range. Iterators increment for each range. When the current element of the first range is less than the second

according to `__comp`, that element is copied and the iterator advances. If the current element of the second range is less, no element is copied and the iterator advances. If an element is contained in both ranges according to `__comp`, no elements are copied and both ranges advance. The output range may not overlap either input range.

set_intersection() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator >
constexpr _OutputIterator std::set_intersection (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result ) [inline], [constexpr]
```

Return the intersection of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in both ranges in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that iterator advances. If an element is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

set_intersection() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename
_Compare >
constexpr _OutputIterator std::set_intersection (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp ) [inline], [constexpr]
```

Return the intersection of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in both ranges in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `__comp`, that iterator advances. If an element is contained in both ranges according to `__comp`, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

set_symmetric_difference() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator >
constexpr _OutputIterator std::set_symmetric_difference (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result ) [inline], [constexpr]
```

Return the symmetric difference of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in one range but not the other in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that element is copied and the iterator advances. If an element is contained in both ranges, no elements are copied and both ranges advance. The output range may not overlap either input range.

set_symmetric_difference() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename
_Compare >
constexpr _OutputIterator std::set_symmetric_difference (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp ) [inline], [constexpr]
```

Return the symmetric difference of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in one range but not the other in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `comp`, that element is copied and the iterator advances. If an element is contained in both ranges according to `__comp`, no elements are copied and both ranges advance. The output range may not overlap either input range.

set_union() [1/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator >
constexpr _OutputIterator std::set_union (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result ) [inline], [constexpr]
```

Return the union of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in each range in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that element is copied and the iterator advanced. If an element is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

set_union() [2/2]

```
template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename
_Compare >
constexpr _OutputIterator std::set_union (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp ) [inline], [constexpr]
```

Return the union of two sorted ranges using a comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__result</code>	Start of output range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in each range in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `__comp`, that element is copied and the iterator advanced. If an equivalent element according to `__comp` is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

3.2 Atomics

Collaboration diagram for Atomics:



Classes

- struct `std::__atomic_base< _Tp >`
- struct `std::__atomic_base< _TPp * >`
- struct `std::__atomic_flag_base`
- class `std::atomic< _Tp >`
- struct `std::atomic< _Tp * >`
- struct `std::atomic< bool >`
- struct `std::atomic< char >`
- struct `std::atomic< char16_t >`
- struct `std::atomic< char32_t >`
- struct `std::atomic< int >`
- struct `std::atomic< long >`
- struct `std::atomic< long long >`
- struct `std::atomic< short >`
- struct `std::atomic< signed char >`
- struct `std::atomic< unsigned char >`
- struct `std::atomic< unsigned int >`
- struct `std::atomic< unsigned long >`
- struct `std::atomic< unsigned long long >`
- struct `std::atomic< unsigned short >`
- struct `std::atomic< wchar_t >`
- struct `std::atomic_flag`
- struct `std::atomic_ref< _Tp >`

Macros

- `#define __cpp_lib_atomic_float`
- `#define __cpp_lib_atomic_is_always_lock_free`
- `#define __cpp_lib_atomic_ref`
- `#define __cpp_lib_atomic_value_initialization`
- `#define _GLIBCXX20_INIT(I)`
- `#define _GLIBCXX20_INIT(I)`
- `#define ATOMIC_BOOL_LOCK_FREE`
- `#define ATOMIC_CHAR16_T_LOCK_FREE`

- `#define ATOMIC_CHAR32_T_LOCK_FREE`
- `#define ATOMIC_CHAR_LOCK_FREE`
- `#define ATOMIC_FLAG_INIT`
- `#define ATOMIC_INT_LOCK_FREE`
- `#define ATOMIC_LLONG_LOCK_FREE`
- `#define ATOMIC_LONG_LOCK_FREE`
- `#define ATOMIC_POINTER_LOCK_FREE`
- `#define ATOMIC_SHORT_LOCK_FREE`
- `#define ATOMIC_VAR_INIT(_VI)`
- `#define ATOMIC_WCHAR_T_LOCK_FREE`

Typedefs

- `typedef unsigned char std::__atomic_flag_data_type`
- `typedef atomic< bool > std::atomic_bool`
- `typedef atomic< char > std::atomic_char`
- `typedef atomic< char16_t > std::atomic_char16_t`
- `typedef atomic< char32_t > std::atomic_char32_t`
- `typedef atomic< int > std::atomic_int`
- `typedef atomic< int16_t > std::atomic_int16_t`
- `typedef atomic< int32_t > std::atomic_int32_t`
- `typedef atomic< int64_t > std::atomic_int64_t`
- `typedef atomic< int8_t > std::atomic_int8_t`
- `typedef atomic< int_fast16_t > std::atomic_int_fast16_t`
- `typedef atomic< int_fast32_t > std::atomic_int_fast32_t`
- `typedef atomic< int_fast64_t > std::atomic_int_fast64_t`
- `typedef atomic< int_fast8_t > std::atomic_int_fast8_t`
- `typedef atomic< int_least16_t > std::atomic_int_least16_t`
- `typedef atomic< int_least32_t > std::atomic_int_least32_t`
- `typedef atomic< int_least64_t > std::atomic_int_least64_t`
- `typedef atomic< int_least8_t > std::atomic_int_least8_t`
- `typedef atomic< intmax_t > std::atomic_intmax_t`
- `typedef atomic< intptr_t > std::atomic_intptr_t`
- `typedef atomic< long long > std::atomic_llong`
- `typedef atomic< long > std::atomic_long`
- `typedef atomic< ptrdiff_t > std::atomic_ptrdiff_t`
- `typedef atomic< signed char > std::atomic_schar`
- `typedef atomic< short > std::atomic_short`
- `typedef atomic< size_t > std::atomic_size_t`
- `typedef atomic< unsigned char > std::atomic_uchar`
- `typedef atomic< unsigned int > std::atomic_uint`
- `typedef atomic< uint16_t > std::atomic_uint16_t`
- `typedef atomic< uint32_t > std::atomic_uint32_t`
- `typedef atomic< uint64_t > std::atomic_uint64_t`
- `typedef atomic< uint8_t > std::atomic_uint8_t`
- `typedef atomic< uint_fast16_t > std::atomic_uint_fast16_t`
- `typedef atomic< uint_fast32_t > std::atomic_uint_fast32_t`
- `typedef atomic< uint_fast64_t > std::atomic_uint_fast64_t`
- `typedef atomic< uint_fast8_t > std::atomic_uint_fast8_t`
- `typedef atomic< uint_least16_t > std::atomic_uint_least16_t`

- typedef `atomic< uint_least32_t >` `std::atomic_uint_least32_t`
- typedef `atomic< uint_least64_t >` `std::atomic_uint_least64_t`
- typedef `atomic< uint_least8_t >` `std::atomic_uint_least8_t`
- typedef `atomic< uintmax_t >` `std::atomic_uintmax_t`
- typedef `atomic< uintptr_t >` `std::atomic_uintptr_t`
- typedef `atomic< unsigned long long >` `std::atomic_ullong`
- typedef `atomic< unsigned long >` `std::atomic_ulong`
- typedef `atomic< unsigned short >` `std::atomic_ushort`
- typedef `atomic< wchar_t >` `std::atomic_wchar_t`

Enumerations

- enum `__memory_order_modifier` { `__memory_order_mask` , `__memory_order_modifier_mask` , `__memory_order_hle_acquire` , `__memory_order_hle_release` }
- enum class `std::memory_order` : int { `relaxed` , `consume` , `acquire` , `release` , `acq_rel` , `seq_cst` }

Functions

- constexpr `memory_order` `std::__cmpexch_failure_order` (`memory_order` __m) noexcept
- constexpr `memory_order` `std::__cmpexch_failure_order2` (`memory_order` __m) noexcept
- constexpr bool `std::__is_valid_cmpexch_failure_order` (`memory_order` __m) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_strong` (`atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_strong` (volatile `atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_strong_explicit` (`atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2, `memory_order` __m1, `memory_order` __m2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_strong_explicit` (volatile `atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2, `memory_order` __m1, `memory_order` __m2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_weak` (`atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_weak` (volatile `atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_weak_explicit` (`atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2, `memory_order` __m1, `memory_order` __m2) noexcept
- template<typename _ITp >
bool `std::atomic_compare_exchange_weak_explicit` (volatile `atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` *__i1, `__atomic_val_t< _ITp >` __i2, `memory_order` __m1, `memory_order` __m2) noexcept
- template<typename _ITp >
`_ITp` `std::atomic_exchange` (`atomic< _ITp >` *__a, `__atomic_val_t< _ITp >` __i) noexcept

- `template<typename _ITp >`
`_ITp std::atomic_exchange (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, ←`
`memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`

- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `void std::atomic_flag_clear (atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_notify_all (atomic_flag *__a) noexcept`
- `void std::atomic_flag_notify_one (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test (const atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test (const volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_explicit (const atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_explicit (const volatile atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_wait (atomic_flag *__a, bool __old) noexcept`
- `void std::atomic_flag_wait_explicit (atomic_flag *__a, bool __old, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const volatile atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const volatile atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const volatile atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::atomic_notify_all (atomic< _Tp > *__a) noexcept`
- `template<typename _Tp >`
`void std::atomic_notify_one (atomic< _Tp > *__a) noexcept`
- `void std::atomic_signal_fence (memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`

- `template<typename _ITp >`
`void std::atomic_store_explicit (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `void std::atomic_thread_fence (memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::atomic_wait (const atomic< _Tp > *__a, typename std::atomic< _Tp >::value_type __old) noexcept`
- `template<typename _Tp >`
`void std::atomic_wait_explicit (const atomic< _Tp > *__a, typename std::atomic< _Tp >::value_type __old, std::memory_order __m) noexcept`
- `template<typename _Tp >`
`_Tp std::kill_dependency (_Tp __y) noexcept`
- `constexpr memory_order std::operator& (memory_order __m, __memory_order_modifier __mod)`
- `constexpr memory_order std::operator| (memory_order __m, __memory_order_modifier __mod)`

Variables

- `constexpr memory_order std::memory_order_acq_rel`
- `constexpr memory_order std::memory_order_acquire`
- `constexpr memory_order std::memory_order_consume`
- `constexpr memory_order std::memory_order_relaxed`
- `constexpr memory_order std::memory_order_release`
- `constexpr memory_order std::memory_order_seq_cst`

3.2.1 Detailed Description

Components for performing atomic operations.

3.2.2 Macro Definition Documentation

ATOMIC_BOOL_LOCK_FREE

```
#define ATOMIC_BOOL_LOCK_FREE
```

Lock-free property.

0 indicates that the types are never lock-free. 1 indicates that the types are sometimes lock-free. 2 indicates that the types are always lock-free.

3.2.3 Typedef Documentation

atomic_bool

```
typedef atomic<bool> std::atomic\_bool
```

[atomic_bool](#)

atomic_char

```
typedef atomic<char> std::atomic_char
```

atomic_char

atomic_char16_t

```
typedef atomic<char16_t> std::atomic_char16_t
```

atomic_char16_t

atomic_char32_t

```
typedef atomic<char32_t> std::atomic_char32_t
```

atomic_char32_t

atomic_int

```
typedef atomic<int> std::atomic_int
```

atomic_int

atomic_int16_t

```
typedef atomic<int16_t> std::atomic_int16_t
```

atomic_int16_t

atomic_int32_t

```
typedef atomic<int32_t> std::atomic_int32_t
```

atomic_int32_t

atomic_int64_t

```
typedef atomic<int64_t> std::atomic_int64_t
```

atomic_int64_t

atomic_int8_t

```
typedef atomic<int8_t> std::atomic_int8_t
```

atomic_int8_t

atomic_int_fast16_t

```
typedef atomic<int_fast16_t> std::atomic_int_fast16_t
```

atomic_int_fast16_t

atomic_int_fast32_t

```
typedef atomic<int_fast32_t> std::atomic_int_fast32_t
```

atomic_int_fast32_t

atomic_int_fast64_t

```
typedef atomic<int_fast64_t> std::atomic_int_fast64_t
```

atomic_int_fast64_t

atomic_int_fast8_t

```
typedef atomic<int_fast8_t> std::atomic_int_fast8_t
```

atomic_int_fast8_t

atomic_int_least16_t

```
typedef atomic<int_least16_t> std::atomic_int_least16_t
```

atomic_int_least16_t

atomic_int_least32_t

```
typedef atomic<int_least32_t> std::atomic_int_least32_t
```

atomic_int_least32_t

atomic_int_least64_t

```
typedef atomic<int_least64_t> std::atomic_int_least64_t
```

atomic_int_least64_t

atomic_int_least8_t

```
typedef atomic<int_least8_t> std::atomic_int_least8_t
```

atomic_int_least8_t

atomic_intmax_t

```
typedef atomic<intmax_t> std::atomic_intmax_t
```

atomic_intmax_t

atomic_intptr_t

```
typedef atomic<intptr_t> std::atomic_intptr_t
```

atomic_intptr_t

atomic_llong

```
typedef atomic<long long> std::atomic_llong
```

atomic_llong

atomic_long

```
typedef atomic<long> std::atomic_long
```

atomic_long

atomic_ptrdiff_t

```
typedef atomic<ptrdiff_t> std::atomic_ptrdiff_t
```

atomic_ptrdiff_t

atomic_schar

```
typedef atomic<signed char> std::atomic\_schar
```

[atomic_schar](#)

atomic_short

```
typedef atomic<short> std::atomic\_short
```

[atomic_short](#)

atomic_size_t

```
typedef atomic<size_t> std::atomic\_size\_t
```

[atomic_size_t](#)

atomic_uchar

```
typedef atomic<unsigned char> std::atomic\_uchar
```

[atomic_uchar](#)

atomic_uint

```
typedef atomic<unsigned int> std::atomic\_uint
```

[atomic_uint](#)

atomic_uint16_t

```
typedef atomic<uint16_t> std::atomic\_uint16\_t
```

[atomic_uint16_t](#)

atomic_uint32_t

```
typedef atomic<uint32_t> std::atomic\_uint32\_t
```

[atomic_uint32_t](#)

atomic_uint64_t

```
typedef atomic<uint64_t> std::atomic_uint64_t
```

atomic_uint64_t

atomic_uint8_t

```
typedef atomic<uint8_t> std::atomic_uint8_t
```

atomic_uint8_t

atomic_uint_fast16_t

```
typedef atomic<uint_fast16_t> std::atomic_uint_fast16_t
```

atomic_uint_fast16_t

atomic_uint_fast32_t

```
typedef atomic<uint_fast32_t> std::atomic_uint_fast32_t
```

atomic_uint_fast32_t

atomic_uint_fast64_t

```
typedef atomic<uint_fast64_t> std::atomic_uint_fast64_t
```

atomic_uint_fast64_t

atomic_uint_fast8_t

```
typedef atomic<uint_fast8_t> std::atomic_uint_fast8_t
```

atomic_uint_fast8_t

atomic_uint_least16_t

```
typedef atomic<uint_least16_t> std::atomic_uint_least16_t
```

atomic_uint_least16_t

atomic_uint_least32_t

```
typedef atomic<uint_least32_t> std::atomic\_uint\_least32\_t
```

[atomic_uint_least32_t](#)

atomic_uint_least64_t

```
typedef atomic<uint_least64_t> std::atomic\_uint\_least64\_t
```

[atomic_uint_least64_t](#)

atomic_uint_least8_t

```
typedef atomic<uint_least8_t> std::atomic\_uint\_least8\_t
```

[atomic_uint_least8_t](#)

atomic_uintmax_t

```
typedef atomic<uintmax_t> std::atomic\_uintmax\_t
```

[atomic_uintmax_t](#)

atomic_uintptr_t

```
typedef atomic<uintptr_t> std::atomic\_uintptr\_t
```

[atomic_uintptr_t](#)

atomic_ullong

```
typedef atomic<unsigned long long> std::atomic\_ullong
```

[atomic_ullong](#)

atomic_ulong

```
typedef atomic<unsigned long> std::atomic\_ulong
```

[atomic_ulong](#)

atomic_ushort

```
typedef atomic<unsigned short> std::atomic_ushort  
atomic_ushort
```

atomic_wchar_t

```
typedef atomic<wchar_t> std::atomic_wchar_t  
atomic_wchar_t
```

3.2.4 Enumeration Type Documentation**memory_order**

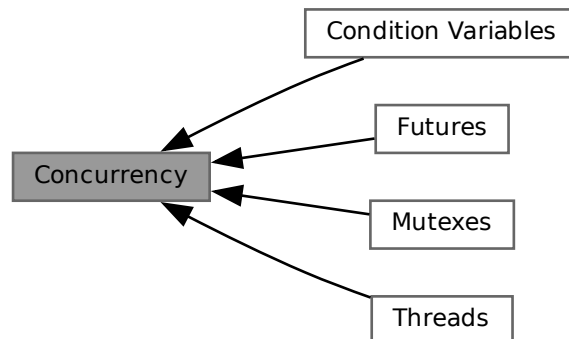
```
enum class std::memory_order : int [strong]  
Enumeration for memory_order.
```

3.2.5 Function Documentation**kill_dependency()**

```
template<typename _Tp >  
_Tp std::kill_dependency (   
    _Tp __y ) [inline], [noexcept]  
kill_dependency
```

3.3 Concurrency

Collaboration diagram for Concurrency:



Modules

- [Condition Variables](#)
- [Futures](#)
- [Mutexes](#)
- [Threads](#)

3.3.1 Detailed Description

Components for concurrent operations, including threads, mutexes, and condition variables.

3.3.2 Condition Variables

Collaboration diagram for Condition Variables:



Classes

- class [std::condition_variable](#)
- class [std::_V2::condition_variable_any](#)

Enumerations

- enum class [std::cv_status](#) { **no_timeout** , **timeout** }

Functions

- void **std::notify_all_at_thread_exit** ([condition_variable](#) &, [unique_lock](#)< [mutex](#) >)

3.3.2.1 Detailed Description

Classes for `condition_variable` support.

3.3.2.2 Enumeration Type Documentation

cv_status

```
enum class std::cv_status [strong]
```

cv_status

3.3.3 Futures

Collaboration diagram for Futures:



Classes

- class `std::__basic_future<_Res>`
- class `std::future<_Res>`
- class `std::future<_Res &>`
- class `std::future<void>`
- class `std::future_error`
- struct `std::is_error_code_enum<future_errc>`
- class `std::packaged_task<_Res(_ArgTypes...)>`
- class `std::promise<_Res>`
- class `std::promise<_Res &>`
- class `std::promise<void>`
- class `std::shared_future<_Res>`
- class `std::shared_future<_Res &>`
- class `std::shared_future<void>`

Enumerations

- enum class `std::future_errc` { `future_already_retrieved` , `promise_already_satisfied` , `no_state` , `broken_←
promise` }
- enum class `std::future_status` { `ready` , `timeout` , `deferred` }
- enum class `std::launch` { `async` , `deferred` }

Functions

- `std::__basic_future< _Res >::__basic_future` (const `shared_future< _Res >` &) noexcept
- `std::__basic_future< _Res >::__basic_future` (`future< _Res >` &&) noexcept
- `std::__basic_future< _Res >::__basic_future` (`shared_future< _Res >` &&) noexcept
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > std::async` (`_Fn` && `_fn`, `_Args` &&... `__args`)
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > std::async` (`launch` `__policy`, `_Fn` && `_fn`, `_Args` &&... `__args`)
- `const error_category & std::future_category` () noexcept
- `error_code std::make_error_code` (`future_errc` `__errc`) noexcept
- `error_condition std::make_error_condition` (`future_errc` `__errc`) noexcept
- `constexpr launch std::operator&` (`launch` `__x`, `launch` `__y`) noexcept
- `launch & std::operator&=` (`launch` & `__x`, `launch` `__y`) noexcept
- `constexpr launch std::operator^` (`launch` `__x`, `launch` `__y`) noexcept
- `launch & std::operator^=` (`launch` & `__x`, `launch` `__y`) noexcept
- `constexpr launch std::operator|` (`launch` `__x`, `launch` `__y`) noexcept
- `launch & std::operator|=` (`launch` & `__x`, `launch` `__y`) noexcept
- `constexpr launch std::operator~` (`launch` `__x`) noexcept
- `template<typename _Fun, typename _Signature = typename __function_guide_helper<decltype(&_Fun::operator())>::type>`
`std::packaged_task` (`_Fun`) -> `packaged_task< _Signature >`
- `template<typename _Res, typename... _ArgTypes>`
`std::packaged_task` (`_Res` (*) (`_ArgTypes...`) -> `packaged_task< _Res(_ArgTypes...)>`
- `shared_future< _Res > std::future< _Res >::share` () noexcept
- `shared_future< _Res & > std::future< _Res & >::share` () noexcept
- `shared_future< void > std::future< void >::share` () noexcept
- `template<typename _Res, typename... _ArgTypes>`
`void std::swap` (`packaged_task< _Res(_ArgTypes...)>` & `__x`, `packaged_task< _Res(_ArgTypes...)>` & `__y`) noexcept
- `template<typename _Res >`
`void std::swap` (`promise< _Res >` & `__x`, `promise< _Res >` & `__y`) noexcept

3.3.3.1 Detailed Description

Futures and promises provide support for retrieving the result from an asynchronous function, e.g. one that is running in another thread. A `std::future` represents an asynchronous result that will become ready at some later time. A consumer can wait on a future until the result is ready to be accessed.

Since

C++11

3.3.3.2 Enumeration Type Documentation

future_errc

```
enum class std::future_errc [strong]
```

Error code for futures.

future_status

```
enum class std::future\_status [strong]
```

Status code for futures.

launch

```
enum class std::launch [strong]
```

Launch code for futures.

3.3.3.3 Function Documentation**async() [1/2]**

```
template<typename _Fn , typename... _Args>
future< __async_result_of< _Fn, _Args... > > std::async (
    _Fn && __fn,
    _Args &&... __args ) [inline]
```

async, potential overload

async() [2/2]

```
template<typename _Fn , typename... _Args>
future< __async_result_of< _Fn, _Args... > > std::async (
    launch __policy,
    _Fn && __fn,
    _Args &&... __args )
```

async

future_category()

```
const error\_category & std::future\_category ( ) [noexcept]
```

Points to a statically-allocated object derived from `error_category`.

make_error_code()

```
error\_code std::make\_error\_code (
    future\_errc __errc ) [inline], [noexcept]
```

Overload of `make_error_code` for `future_errc`.

Referenced by [std::filesystem::filesystem_error::what\(\)](#).

make_error_condition()

```
error_condition std::make_error_condition (
    future_errc __errc ) [inline], [noexcept]
```

Overload of make_error_condition for future_errc.

swap()

```
template<typename _Res , typename... _ArgTypes>
void std::swap (
    packaged_task< _Res (_ArgTypes...)> & __x,
    packaged_task< _Res (_ArgTypes...)> & __y ) [inline], [noexcept]
```

swap

3.3.4 Mutexes

Collaboration diagram for Mutexes:



Classes

- struct [std::adopt_lock_t](#)
- struct [std::defer_lock_t](#)
- class [std::lock_guard< _Mutex >](#)
- class [std::mutex](#)
- struct [std::once_flag](#)
- class [std::recursive_mutex](#)
- class [std::recursive_timed_mutex](#)
- class [std::scoped_lock< _MutexTypes >](#)
- class [std::shared_lock< _Mutex >](#)
- class [std::shared_mutex](#)
- class [std::shared_timed_mutex](#)
- class [std::timed_mutex](#)
- struct [std::try_to_lock_t](#)
- class [std::unique_lock< _Mutex >](#)

Macros

- `#define __cpp_lib_scoped_lock`
- `#define __cpp_lib_shared_mutex`
- `#define __cpp_lib_shared_timed_mutex`

Functions

- `void std::__once_proxy (void)`
- `template<typename _Callable, typename... _Args>`
`void std::call_once (once_flag &__once, _Callable &&__f, _Args &&... __args)`
- `template<typename _L1, typename _L2, typename... _L3>`
`void std::lock (_L1 &__l1, _L2 &__l2, _L3 &... __l3)`
- `template<typename _Mutex >`
`void swap (shared_lock< _Mutex > &__x, shared_lock< _Mutex > &__y) noexcept`
- `template<typename _L1, typename _L2, typename... _L3>`
`int std::try_lock (_L1 &__l1, _L2 &__l2, _L3 &... __l3)`

Variables

- `constexpr adopt_lock_t std::adopt_lock`
- `constexpr defer_lock_t std::defer_lock`
- `constexpr try_to_lock_t std::try_to_lock`

3.3.4.1 Detailed Description

Classes for mutex support.

3.3.4.2 Function Documentation

call_once()

```
template<typename _Callable, typename... _Args>
void std::call_once (
    once_flag & __once,
    _Callable && __f,
    _Args &&... __args )
```

Invoke a callable and synchronize with other calls using the same flag.

lock()

```
template<typename _L1, typename _L2, typename... _L3>
void std::lock (
    _L1 & __l1,
    _L2 & __l2,
    _L3 &... __l3 )
```

Generic lock.

Parameters

\leftrightarrow _l1	Meets Lockable requirements (try_lock() may throw).
\leftrightarrow _l2	Meets Lockable requirements (try_lock() may throw).
\leftrightarrow _l3	Meets Lockable requirements (try_lock() may throw).

Exceptions

<i>An</i>	exception thrown by an argument's lock() or try_lock() member.
-----------	--

Postcondition

All arguments are locked.

All arguments are locked via a sequence of calls to lock(), try_lock() and unlock(). If this function exits via an exception any locks that were obtained will be released.

swap()

```
template<typename _Mutex >
void swap (
    shared_lock< _Mutex > & __x,
    shared_lock< _Mutex > & __y ) [related]
```

Swap specialization for shared_lock.

try_lock()

```
template<typename _L1 , typename _L2 , typename... _L3>
int std::try_lock (
    _L1 & __l1,
    _L2 & __l2,
    _L3 &... __l3 ) [inline]
```

Generic try_lock.

Parameters

\leftrightarrow _l1	Meets Lockable requirements (try_lock() may throw).
\leftrightarrow _l2	Meets Lockable requirements (try_lock() may throw).
\leftrightarrow _l3	Meets Lockable requirements (try_lock() may throw).

Returns

Returns -1 if all `try_lock()` calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls `try_lock()` on each argument.

3.3.4.3 Variable Documentation**adopt_lock**

```
constexpr adopt_lock_t std::adopt_lock [inline], [constexpr]
```

Tag used to make a scoped lock take ownership of a locked mutex.

defer_lock

```
constexpr defer_lock_t std::defer_lock [inline], [constexpr]
```

Tag used to prevent a scoped lock from acquiring ownership of a mutex.

try_to_lock

```
constexpr try_to_lock_t std::try_to_lock [inline], [constexpr]
```

Tag used to prevent a scoped lock from blocking if a mutex is locked.

3.3.5 Threads

Collaboration diagram for Threads:



Namespaces

- namespace `std::this_thread`

Classes

- struct `std::hash< thread::id >`
- class `std::thread::id`
- class `std::jthread`
- class `std::thread`

Functions

- bool **operator==** (`thread::id __x`, `thread::id __y`) noexcept
- void **swap** (`thread &__x`, `thread &__y`) noexcept
- bool **operator!=** (`thread::id __x`, `thread::id __y`) noexcept
- bool **operator<** (`thread::id __x`, `thread::id __y`) noexcept
- bool **operator<=** (`thread::id __x`, `thread::id __y`) noexcept
- bool **operator>** (`thread::id __x`, `thread::id __y`) noexcept
- bool **operator>=** (`thread::id __x`, `thread::id __y`) noexcept
- template<class `_CharT`, class `_Traits` >
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, thread::id __id)`

3.3.5.1 Detailed Description

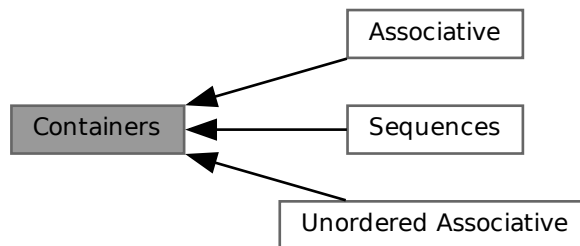
Since

C++11

Classes for thread support.

3.4 Containers

Collaboration diagram for Containers:



Modules

- [Associative](#)
- [Sequences](#)
- [Unordered Associative](#)

3.4.1 Detailed Description

Containers are collections of objects.

A container may hold any type which meets certain requirements, but the type of contained object is chosen at compile time, and all objects in a given container must be of the same type. (Polymorphism is possible by declaring a container of pointers to a base class and then populating it with pointers to instances of derived classes. Variant value types such as the `any` class from `Boost` can also be used.

All contained types must be `Assignable` and `CopyConstructible`. Specific containers may place additional requirements on the types of their contained objects.

Containers manage memory allocation and deallocation themselves when storing your objects. The objects are destroyed when the container is itself destroyed. Note that if you are storing pointers in a container, `delete` is *not* automatically called on the pointers before destroying them.

All containers must meet certain requirements, summarized in `tables`.

The standard containers are further refined into [Sequences](#) and [Associative Containers](#). [Unordered Associative Containers](#).

3.4.2 Associative

Collaboration diagram for Associative:



Modules

- [Node handles](#)

Classes

- class `std::map<_Key, _Tp, _Compare, _Alloc>`
- class `std::multimap<_Key, _Tp, _Compare, _Alloc>`
- class `std::multiset<_Key, _Compare, _Alloc>`
- class `std::set<_Key, _Compare, _Alloc>`

3.4.2.1 Detailed Description

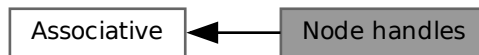
Associative containers allow fast retrieval of data based on keys.

Each container type is parameterized on a `Key` type, and an ordering relation used to sort the elements of the container.

All associative containers must meet certain requirements, summarized in [tables](#).

3.4.2.2 Node handles

Collaboration diagram for Node handles:



Classes

- class `std::_Node_handle<_Key, _Value, _NodeAlloc >`
- class `std::_Node_handle<_Value, _Value, _NodeAlloc >`
- class `std::_Node_handle_common<_Val, _NodeAlloc >`
- struct `std::_Node_insert_return<_Iterator, _NodeHandle >`

3.4.2.2.1 Detailed Description

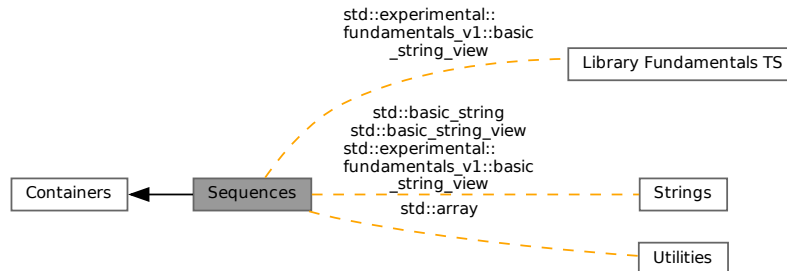
Since

C++17

The associative containers (`map`, `set`, `multimap` and `multiset`) support extracting and re-inserting nodes from the container. Those operations use the container's `node_handle` type, which is an alias for a `_Node_handle<...>` type. You should always use the container's `node_handle` type (e.g. `std::set<int>::node_handle`) to refer to these types, not the non-standard internal `_Node_handle` names.

3.4.3 Sequences

Collaboration diagram for Sequences:



Classes

- struct `std::array<_Tp, _Nm>`
- class `std::basic_string<_CharT, _Traits, _Alloc>`
- class `std::basic_string_view<_CharT, _Traits>`
- class `std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits>`
- class `std::deque<_Tp, _Alloc>`
- class `std::forward_list<_Tp, _Alloc>`
- class `std::list<_Tp, _Alloc>`
- class `std::priority_queue<_Tp, _Sequence, _Compare>`
- class `std::queue<_Tp, _Sequence>`
- class `std::stack<_Tp, _Sequence>`
- class `std::vector<_Tp, _Alloc>`
- class `std::vector<bool, _Alloc>`

3.4.3.1 Detailed Description

Sequences arrange a collection of objects into a strictly linear order.

The differences between sequences are usually due to one or both of the following:

- memory management
- algorithmic complexity

As an example of the first case, `vector` is required to use a contiguous memory layout, while other sequences such as `deque` are not.

The prime reason for choosing one sequence over another should be based on the second category of differences, algorithmic complexity. For example, if you need to perform many inserts and removals from the middle of a sequence, `list` would be ideal. But if you need to perform constant-time access to random elements of the sequence, then `list` should not be used.

All sequences must meet certain requirements, summarized in [tables](#).

3.4.4 Unordered Associative

Collaboration diagram for Unordered Associative:



Classes

- class [std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >](#)
- class [std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >](#)
- class [std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >](#)
- class [std::unordered_set< _Value, _Hash, _Pred, _Alloc >](#)

3.4.4.1 Detailed Description

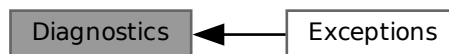
Unordered associative containers allow fast retrieval of data based on keys.

Each container type is parameterized on a `Key` type, a `Hash` type providing a hashing functor, and an ordering relation used to sort the elements of the container.

All unordered associative containers must meet certain requirements, summarized in [tables](#).

3.5 Diagnostics

Collaboration diagram for Diagnostics:



Modules

- [Exceptions](#)

Classes

- class `std::_V2::error_category`
- class `std::error_code`
- class `std::error_condition`
- struct `std::is_error_code_enum< _Tp >`
- struct `std::is_error_condition_enum< _Tp >`

Functions

- `const error_category & std::generic_category ()` noexcept
- `error_code make_error_code (errc __e)` noexcept
- `error_condition make_error_condition (errc __e)` noexcept
- `bool operator!= (const error_code &__lhs, const error_code &__rhs)` noexcept
- `bool operator!= (const error_code &__lhs, const error_condition &__rhs)` noexcept
- `bool operator!= (const error_condition &__lhs, const error_code &__rhs)` noexcept
- `bool operator!= (const error_condition &__lhs, const error_condition &__rhs)` noexcept
- `bool operator< (const error_code &__lhs, const error_code &__rhs)` noexcept
- `bool operator< (const error_condition &__lhs, const error_condition &__rhs)` noexcept
- `template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const error_code &__e)`
- `bool operator== (const error_code &__lhs, const error_code &__rhs)` noexcept
- `bool operator== (const error_code &__lhs, const error_condition &__rhs)` noexcept
- `bool operator== (const error_condition &__lhs, const error_code &__rhs)` noexcept
- `bool operator== (const error_condition &__lhs, const error_condition &__rhs)` noexcept
- `const error_category & std::system_category ()` noexcept

Variables

- `template<typename _Tp > constexpr bool std::is_error_code_enum_v`
- `template<typename _Tp > constexpr bool std::is_error_condition_enum_v`

3.5.1 Detailed Description

Components for error handling, reporting, and diagnostic operations.

3.5.2 Function Documentation

`generic_category()`

```
const error_category & std::_V2::generic_category ( ) [noexcept]
```

Error category for `errno` error codes.

make_error_code()

```
error_code make_error_code (
    errc __e ) [related]
```

Create an `error_code` representing a standard `errc` condition.

The `std::errc` constants correspond to `errno` macros and so use the generic category.

Since

C++11

make_error_condition()

```
error_condition make_error_condition (
    errc __e ) [related]
```

Create an `error_condition` representing a standard `errc` condition.

The `std::errc` constants correspond to `errno` macros and so use the generic category.

Since

C++11

operator<() [1/2]

```
bool operator< (
    const error_code & __lhs,
    const error_code & __rhs ) [related]
```

Ordered comparison for `std::error_code`.

This defines a total order by comparing the categories, and then if they are equal comparing the values.

Since

C++11

operator<() [2/2]

```
bool operator< (
    const error_condition & __lhs,
    const error_condition & __rhs ) [related]
```

Ordered comparison for `std::error_condition`.

This defines a total order by comparing the categories, and then if they are equal comparing the values.

Since

C++11

operator<<()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & operator<< (
    basic_ostream< _CharT, _Traits > & __os,
    const error_code & __e ) [related]
```

Write a `std::error_code` to an ostream.

Since

C++11

operator==() [1/3]

```
bool operator== (
    const error_code & __lhs,
    const error_code & __rhs ) [related]
```

Equality comparison for `std::error_code`.

Returns true only if they have the same category and the same value.

Since

C++11

operator==() [2/3]

```
bool operator== (
    const error_code & __lhs,
    const error_condition & __rhs ) [related]
```

Equality comparison for `std::error_code` and `std::error_condition`.

Uses each category's `equivalent` member function to check whether the values correspond to an equivalent error in that category.

Since

C++11

operator==() [3/3]

```
bool operator== (
    const error_condition & __lhs,
    const error_condition & __rhs ) [related]
```

Equality comparison for `std::error_condition`.

Returns true only if they have the same category and the same value.

Since

C++11

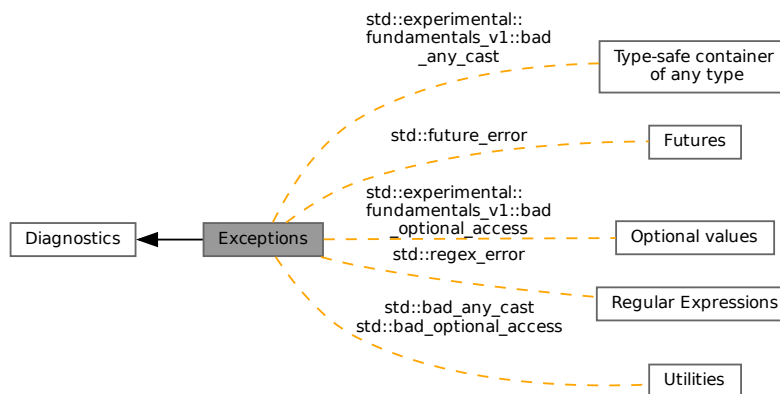
system_category()

```
const error_category & std::_V2::system_category ( ) [noexcept]
```

Error category for other error codes defined by the OS.

3.5.3 Exceptions

Collaboration diagram for Exceptions:



Classes

- class `__cxxabiv1::__forced_unwind`
- class `std::bad_alloc`
- class `std::bad_any_cast`
- class `std::experimental::fundamentals_v1::bad_any_cast`
- class `std::bad_cast`
- class `std::bad_exception`
- class `std::bad_function_call`
- class `std::experimental::fundamentals_v1::bad_optional_access`
- class `std::bad_optional_access`
- class `std::bad_typeid`
- class `std::bad_weak_ptr`
- class `std::domain_error`
- class `std::exception`
- class `std::__exception_ptr::exception_ptr`
- struct `__gnu_cxx::forced_error`
- class `std::future_error`
- class `std::invalid_argument`
- class `std::length_error`
- class `std::logic_error`
- class `std::nested_exception`
- class `std::out_of_range`
- class `std::overflow_error`
- class `std::range_error`
- class `__gnu_cxx::recursive_init_error`
- class `std::regex_error`
- class `std::runtime_error`
- class `std::system_error`
- class `std::underflow_error`

Macros

- `#define __cpp_lib_uncaught_exceptions`

Typedefs

- typedef void(* `std::terminate_handler`) ()
- typedef void(* `std::unexpected_handler`) ()

Functions

- `void __gnu_cxx::__verbose_terminate_handler ()`
- `exception_ptr std::current_exception () noexcept`
- `terminate_handler std::get_terminate () noexcept`
- `unexpected_handler std::get_unexpected () noexcept`
- `template<typename _Ex >`
`exception_ptr std::make_exception_ptr (_Ex __ex) noexcept`
- `void std::rethrow_exception (exception_ptr)`
- `template<typename _Ex >`
`void std::rethrow_if_nested (const _Ex &__ex)`
- `terminate_handler std::set_terminate (terminate_handler) noexcept`
- `unexpected_handler std::set_unexpected (unexpected_handler) noexcept`
- `void std::terminate () noexcept`
- `template<typename _Tp >`
`void std::throw_with_nested (_Tp &&__t)`
- `bool std::uncaught_exception () noexcept`
- `int std::uncaught_exceptions () noexcept`
- `void std::unexpected ()`

3.5.3.1 Detailed Description

Since

C++98

Classes and functions for reporting errors via exceptions.

3.5.3.2 Typedef Documentation

terminate_handler

```
typedef void(* std::terminate_handler) ()
```

If you write a replacement terminate handler, it must be of this type.

unexpected_handler

```
typedef void(* std::unexpected_handler) ()
```

If you write a replacement unexpected handler, it must be of this type.

3.5.3.3 Function Documentation

`__verbose_terminate_handler()`

```
void __gnu_cxx::__verbose_terminate_handler ( )
```

A replacement for the standard `terminate_handler` which prints more information about the terminating exception (if any) on `stderr`.

Call

```
std::set_terminate(__gnu_cxx::__verbose_terminate_handler)
```

to use. For more info, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/bk01pt02ch06s02.html>.

In 3.4 and later, this is on by default.

References [__gnu_cxx::__verbose_terminate_handler\(\)](#).

Referenced by [__gnu_cxx::__verbose_terminate_handler\(\)](#).

`current_exception()`

```
exception_ptr std::current_exception ( ) [noexcept]
```

Obtain an `exception_ptr` to the currently handled exception. If there is none, or the currently handled exception is foreign, return the null value.

Referenced by [std::make_exception_ptr\(\)](#).

`get_terminate()`

```
terminate_handler std::get_terminate ( ) [noexcept]
```

Return the current terminate handler.

`get_unexpected()`

```
unexpected_handler std::get_unexpected ( ) [noexcept]
```

Return the current unexpected handler.

Since

C++11

Deprecated Removed from the C++ standard in C++17

make_exception_ptr()

```
template<typename _Ex >
exception_ptr std::make_exception_ptr (
    _Ex __ex ) [noexcept]
```

Obtain an exception_ptr pointing to a copy of the supplied object.

References [std::current_exception\(\)](#).

rethrow_exception()

```
void std::rethrow_exception (
    exception_ptr )
```

Throw the object pointed to by the exception_ptr.

References [std::rethrow_exception\(\)](#).

Referenced by [std::rethrow_exception\(\)](#).

rethrow_if_nested()

```
template<typename _Ex >
void std::rethrow_if_nested (
    const _Ex & __ex ) [inline]
```

If __ex is derived from nested_exception, __ex.rethrow_nested().

References [std::__addressof\(\)](#).

set_terminate()

```
terminate_handler std::set_terminate (
    terminate_handler ) [noexcept]
```

Takes a new handler function as an argument, returns the old function.

set_unexpected()

```
unexpected_handler std::set_unexpected (
    unexpected_handler ) [noexcept]
```

Takes a new handler function as an argument, returns the old function.

Deprecated Removed from the C++ standard in C++17

terminate()

```
void std::terminate ( ) [noexcept]
```

The runtime will call this function if exception handling must be abandoned for any reason. It can also be called by the user.

throw_with_nested()

```
template<typename _Tp >  
void std::throw_with_nested (   
    _Tp && __t ) [inline]
```

If `__t` is derived from `nested_exception`, throws `__t`. Else, throws an implementation-defined object derived from both.

uncaught_exception()

```
bool std::uncaught_exception ( ) [noexcept]
```

[18.6.4]/1: 'Returns true after completing evaluation of a throw-expression until either completing initialization of the exception-declaration in the matching handler or entering `unexpected()` due to the throw; or after entering `terminate()` for any reason other than an explicit call to `terminate()`. [Note: This includes stack unwinding [15.2]. end note]'

2: 'When `uncaught_exception()` is true, throwing an exception can result in a call of `terminate()`' (15.5.1).'

uncaught_exceptions()

```
int std::uncaught_exceptions ( ) [noexcept]
```

The number of uncaught exceptions.

Since

C++17, or any non-strict mode, e.g. `-std=gnu++98`

See also

`uncaught_exception()`

unexpected()

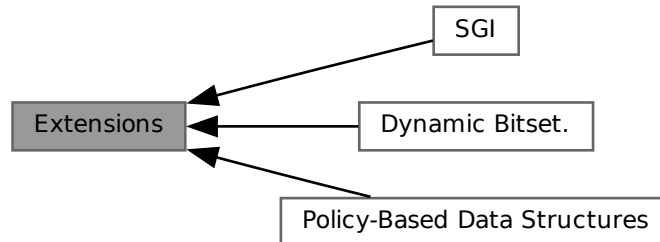
```
void std::unexpected ( )
```

The runtime will call this function if an exception is thrown which violates the function's exception specification.

Deprecated Removed from the C++ standard in C++17

3.6 Extensions

Collaboration diagram for Extensions:



Modules

- [Dynamic Bitset.](#)
- [Policy-Based Data Structures](#)
- [SGI](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std::tr2](#)

Classes

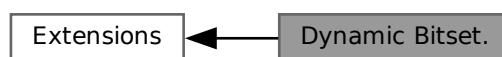
- class [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >](#)

3.6.1 Detailed Description

Components generally useful that are not part of any standard.

3.6.2 Dynamic Bitset.

Collaboration diagram for Dynamic Bitset.:



Classes

- struct `std::tr2::__dynamic_bitset_base<_WordT, _Alloc>`
- class `std::tr2::dynamic_bitset<_WordT, _Alloc>`
- class `std::tr2::dynamic_bitset<_WordT, _Alloc>::reference`

Functions

- template<typename _CharT, typename _Traits, typename _Alloc1>
void `std::tr2::dynamic_bitset<_WordT, _Alloc>::M_copy_to_string` (`std::basic_string<_CharT, _Traits, ↵_Alloc1>` &__str, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1')) const
- template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc>
`std::basic_ostream<_CharT, _Traits>` & `std::tr2::operator<<` (`std::basic_ostream<_CharT, _Traits>` &__os, const `dynamic_bitset<_WordT, _Alloc>` &__x)
- template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc>
`std::basic_istream<_CharT, _Traits>` & `std::tr2::operator>>` (`std::basic_istream<_CharT, _Traits>` &__↵is, `dynamic_bitset<_WordT, _Alloc>` &__x)
- template<typename _WordT, typename _Alloc>
bool `std::tr2::operator!=` (const `dynamic_bitset<_WordT, _Alloc>` &__lhs, const `dynamic_bitset<_WordT, _Alloc>` &__rhs)
- template<typename _WordT, typename _Alloc>
bool `std::tr2::operator<=` (const `dynamic_bitset<_WordT, _Alloc>` &__lhs, const `dynamic_bitset<_WordT, _↵Alloc>` &__rhs)
- template<typename _WordT, typename _Alloc>
bool `std::tr2::operator>` (const `dynamic_bitset<_WordT, _Alloc>` &__lhs, const `dynamic_bitset<_WordT, _Alloc>` &__rhs)
- template<typename _WordT, typename _Alloc>
bool `std::tr2::operator>=` (const `dynamic_bitset<_WordT, _Alloc>` &__lhs, const `dynamic_bitset<_WordT, _↵Alloc>` &__rhs)
- template<typename _WordT, typename _Alloc>
`dynamic_bitset<_WordT, _Alloc>` `std::tr2::operator&` (const `dynamic_bitset<_WordT, _Alloc>` &__x, const `dynamic_bitset<_WordT, _Alloc>` &__y)
- template<typename _WordT, typename _Alloc>
`dynamic_bitset<_WordT, _Alloc>` `std::tr2::operator|` (const `dynamic_bitset<_WordT, _Alloc>` &__x, const `dynamic_bitset<_WordT, _Alloc>` &__y)
- template<typename _WordT, typename _Alloc>
`dynamic_bitset<_WordT, _Alloc>` `std::tr2::operator^` (const `dynamic_bitset<_WordT, _Alloc>` &__x, const `dynamic_bitset<_WordT, _Alloc>` &__y)
- template<typename _WordT, typename _Alloc>
`dynamic_bitset<_WordT, _Alloc>` `std::tr2::operator-` (const `dynamic_bitset<_WordT, _Alloc>` &__x, const `dynamic_bitset<_WordT, _Alloc>` &__y)

3.6.2.1 Detailed Description

3.6.2.2 Function Documentation

operator"!=(())

```
template<typename _WordT , typename _Alloc >
bool std::tr2::operator!= (
    const dynamic_bitset< _WordT, _Alloc > & __lhs,
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

operator&()

```
template<typename _WordT , typename _Alloc >
dynamic_bitset< _WordT, _Alloc > std::tr2::operator& (
    const dynamic_bitset< _WordT, _Alloc > & __x,
    const dynamic_bitset< _WordT, _Alloc > & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

\leftrightarrow __x	A bitset.
\leftrightarrow __y	A bitset of the same size as __x.

Returns

A new bitset.

These should be self-explanatory.

operator-()

```
template<typename _WordT , typename _Alloc >
dynamic_bitset< _WordT, _Alloc > std::tr2::operator- (
    const dynamic_bitset< _WordT, _Alloc > & __x,
    const dynamic_bitset< _WordT, _Alloc > & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

A new bitset.

These should be self-explanatory.

operator<<()

```
template<typename _CharT , typename _Traits , typename _WordT , typename _Alloc >
std::basic_ostream< _CharT, _Traits > & std::tr2::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const dynamic_bitset< _WordT, _Alloc > & __x ) [inline]
```

Stream output operator for dynamic_bitset.

operator<=()

```
template<typename _WordT , typename _Alloc >
bool std::tr2::operator<= (
    const dynamic_bitset< _WordT, _Alloc > & __lhs,
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

operator>()

```
template<typename _WordT , typename _Alloc >
bool std::tr2::operator> (
    const dynamic_bitset< _WordT, _Alloc > & __lhs,
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

operator>=()

```
template<typename _WordT , typename _Alloc >
bool std::tr2::operator>= (
    const dynamic_bitset< _WordT, _Alloc > & __lhs,
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

operator>>()

```
template<typename _CharT , typename _Traits , typename _WordT , typename _Alloc >
std::basic_istream< _CharT, _Traits > & std::tr2::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    dynamic_bitset< _WordT, _Alloc > & __x )
```

Stream input operator for dynamic_bitset.

Input will skip whitespace and only accept '0' and '1' characters. The dynamic_bitset will grow as necessary to hold the string of bits.

References [std::basic_string< _CharT, _Traits, _Alloc >::empty\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::push_back\(\)](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::reserve\(\)](#), [std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#), [std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset\(\)](#) and [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

operator^()

```
template<typename _WordT , typename _Alloc >
dynamic_bitset< _WordT, _Alloc > std::tr2::operator^ (
    const dynamic_bitset< _WordT, _Alloc > & __x,
    const dynamic_bitset< _WordT, _Alloc > & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

\leftarrow __x	A bitset.
\leftarrow __y	A bitset of the same size as __x.

Returns

A new bitset.

These should be self-explanatory.

operator" | ()

```
template<typename _WordT , typename _Alloc >
dynamic_bitset< _WordT, _Alloc > std::tr2::operator| (
    const dynamic_bitset< _WordT, _Alloc > & __x,
    const dynamic_bitset< _WordT, _Alloc > & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

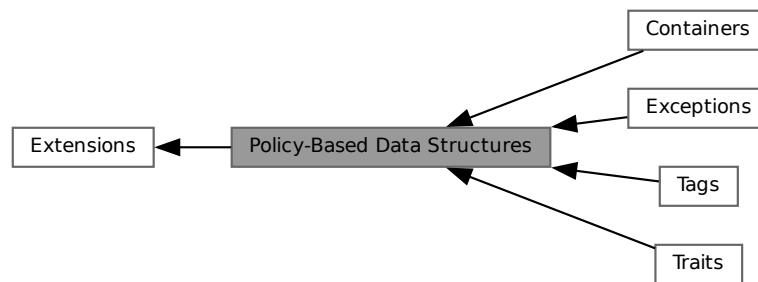
Returns

A new bitset.

These should be self-explanatory.

3.6.3 Policy-Based Data Structures

Collaboration diagram for Policy-Based Data Structures:

**Modules**

- [Containers](#)
- [Exceptions](#)
- [Tags](#)
- [Traits](#)

Classes

- `struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type >`

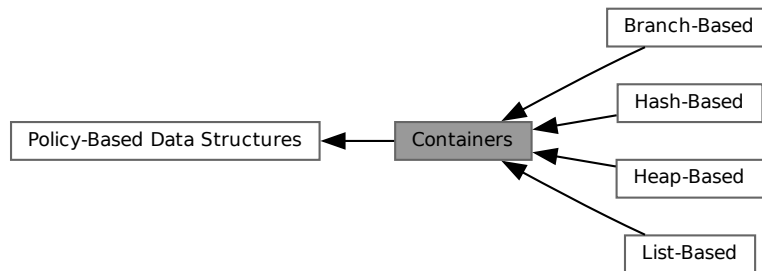
3.6.3.1 Detailed Description

This is a library of policy-based elementary data structures: associative containers and priority queues. It is designed for high-performance, flexibility, semantic safety, and conformance to the corresponding containers in std (except for some points where it differs by design).

For details, see: http://gcc.gnu.org/onlinedocs/libstdc++/ext/pb_ds/index.html

3.6.3.2 Containers

Collaboration diagram for Containers:



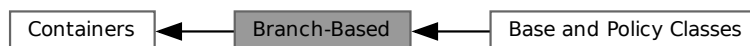
Modules

- [Branch-Based](#)
- [Hash-Based](#)
- [Heap-Based](#)
- [List-Based](#)

3.6.3.2.1 Detailed Description

3.6.3.2.2 Branch-Based

Collaboration diagram for Branch-Based:



Modules

- [Base and Policy Classes](#)

Classes

- class [__gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >](#)
- class [__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >](#)
- class [__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >](#)

Macros

- `#define PB_DS_BRANCH_BASE`
- `#define PB_DS_TREE_BASE`
- `#define PB_DS_TREE_NODE_AND_IT_TRAITS`
- `#define PB_DS_TRIE_BASE`
- `#define PB_DS_TRIE_NODE_AND_IT_TRAITS`

Detailed Description

Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



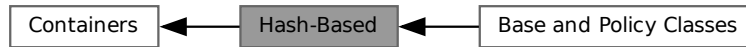
*Classes

- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >](#)
- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)
- class [__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >](#)
- class [__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)
- class [__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)

Detailed Description

3.6.3.2.3 Hash-Based

Collaboration diagram for Hash-Based:



Modules

- [Base and Policy Classes](#)

Classes

- class `__gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >`
- class `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >`
- class `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >`

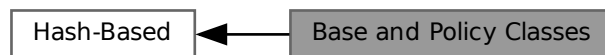
Macros

- `#define PB_DS_CC_HASH_BASE`
- `#define PB_DS_GP_HASH_BASE`
- `#define PB_DS_HASH_BASE`

Detailed Description

Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



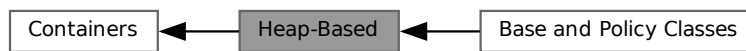
*Classes

- class `__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >`
- class `__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >`

Detailed Description

3.6.3.2.4 Heap-Based

Collaboration diagram for Heap-Based:



Modules

- [Base and Policy Classes](#)

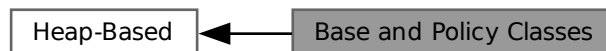
Classes

- [class `__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>`](#)

Detailed Description

Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



*Classes

- [class `__gnu_pbds::detail::binary_heap<Value_Type, Cmp_Fn, _Alloc>`](#)
- [class `__gnu_pbds::detail::binomial_heap<Value_Type, Cmp_Fn, _Alloc>`](#)
- [class `__gnu_pbds::detail::pairing_heap<Value_Type, Cmp_Fn, _Alloc>`](#)
- [class `__gnu_pbds::detail::rc_binomial_heap<Value_Type, Cmp_Fn, _Alloc>`](#)
- [class `__gnu_pbds::detail::thin_heap<Value_Type, Cmp_Fn, _Alloc>`](#)

Detailed Description

3.6.3.2.5 List-Based

Collaboration diagram for List-Based:



Classes

- class [__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >](#)

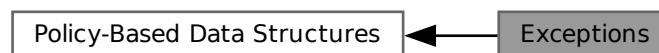
Macros

- `#define PB_DS_LU_BASE`

Detailed Description

3.6.3.3 Exceptions

Collaboration diagram for Exceptions:



Classes

- struct [__gnu_pbds::container_error](#)
- struct [__gnu_pbds::insert_error](#)
- struct [__gnu_pbds::join_error](#)
- struct [__gnu_pbds::resize_error](#)

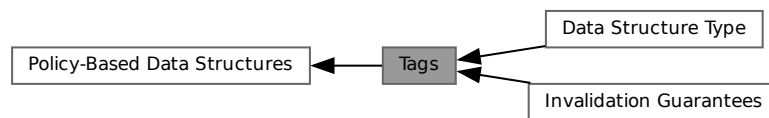
Functions

- void `__gnu_pbds::__throw_container_error()`
- void `__gnu_pbds::__throw_insert_error()`
- void `__gnu_pbds::__throw_join_error()`
- void `__gnu_pbds::__throw_resize_error()`

3.6.3.3.1 Detailed Description

3.6.3.4 Tags

Collaboration diagram for Tags:



Modules

- [Data Structure Type](#)
- [Invalidation Guarantees](#)

Classes

- struct [__gnu_pbds::trivial_iterator_tag](#)

Typedefs

- typedef void [__gnu_pbds::trivial_iterator_difference_type](#)

3.6.3.4.1 Detailed Description

3.6.3.4.2 Typedef Documentation

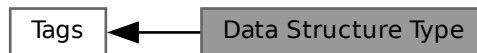
trivial_iterator_difference_type

```
typedef void __gnu_pbds::trivial_iterator_difference_type
```

Prohibit moving trivial iterators.

3.6.3.4.3 Data Structure Type

Collaboration diagram for Data Structure Type:



Classes

- struct [__gnu_pbds::associative_tag](#)
- struct [__gnu_pbds::basic_branch_tag](#)
- struct [__gnu_pbds::basic_hash_tag](#)
- struct [__gnu_pbds::binary_heap_tag](#)
- struct [__gnu_pbds::binomial_heap_tag](#)
- struct [__gnu_pbds::cc_hash_tag](#)
- struct [__gnu_pbds::container_tag](#)
- struct [__gnu_pbds::gp_hash_tag](#)
- struct [__gnu_pbds::list_update_tag](#)
- struct [__gnu_pbds::ov_tree_tag](#)
- struct [__gnu_pbds::pairing_heap_tag](#)
- struct [__gnu_pbds::pat_trie_tag](#)
- struct [__gnu_pbds::priority_queue_tag](#)
- struct [__gnu_pbds::rb_tree_tag](#)
- struct [__gnu_pbds::rc_binomial_heap_tag](#)
- struct [__gnu_pbds::sequence_tag](#)
- struct [__gnu_pbds::splay_tree_tag](#)
- struct [__gnu_pbds::string_tag](#)
- struct [__gnu_pbds::thin_heap_tag](#)
- struct [__gnu_pbds::tree_tag](#)
- struct [__gnu_pbds::trie_tag](#)

Detailed Description

3.6.3.4.4 Invalidation Guarantees

Collaboration diagram for Invalidation Guarantees:



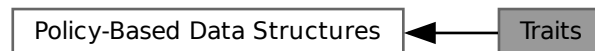
Classes

- struct [__gnu_pbds::basic_invalidation_guarantee](#)
- struct [__gnu_pbds::point_invalidation_guarantee](#)
- struct [__gnu_pbds::range_invalidation_guarantee](#)

Detailed Description

3.6.3.5 Traits

Collaboration diagram for Traits:



Classes

- struct [__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >](#)
- struct [__gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >](#)
- struct [__gnu_pbds::container_traits< Cntnr >](#)
- struct [__gnu_pbds::container_traits_base< _Tag >](#)
- struct [__gnu_pbds::container_traits_base< binary_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< binomial_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< cc_hash_tag >](#)

- struct `__gnu_pbds::container_traits_base< gp_hash_tag >`
- struct `__gnu_pbds::container_traits_base< list_update_tag >`
- struct `__gnu_pbds::container_traits_base< ov_tree_tag >`
- struct `__gnu_pbds::container_traits_base< pairing_heap_tag >`
- struct `__gnu_pbds::container_traits_base< pat_trie_tag >`
- struct `__gnu_pbds::container_traits_base< rb_tree_tag >`
- struct `__gnu_pbds::container_traits_base< rc_binomial_heap_tag >`
- struct `__gnu_pbds::container_traits_base< splay_tree_tag >`
- struct `__gnu_pbds::container_traits_base< thin_heap_tag >`
- struct `__gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::no_throw_copies< Key, Mapped >`
- struct `__gnu_pbds::detail::no_throw_copies< Key, null_type >`
- struct `__gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 >`
- struct `__gnu_pbds::null_type`
- struct `__gnu_pbds::detail::rebind_traits< _Alloc, T >`
- struct `__gnu_pbds::detail::select_value_type< Key, Mapped >`
- struct `__gnu_pbds::detail::select_value_type< Key, null_type >`
- struct `__gnu_pbds::detail::stored_data< _Tv, _Th, Store_Hash >`
- struct `__gnu_pbds::detail::stored_data< _Tv, _Th, false >`
- struct `__gnu_pbds::detail::stored_hash< _Th >`
- struct `__gnu_pbds::detail::stored_value< _Tv >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >`
- struct `__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >`
- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, false >`
- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, true >`
- struct `__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >`
- struct `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >`
- struct `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >`
- struct `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >`

Variables

- static `null_type` `__gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash >::s_null_type`

3.6.3.5.1 Detailed Description

3.6.4 SGI

Collaboration diagram for SGI:



Classes

- class `__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >`
- struct `__gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 >`
- struct `__gnu_cxx::constant_unary_fun< _Result, _Argument >`
- struct `__gnu_cxx::constant_void_fun< _Result >`
- class `__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc >`
- struct `__gnu_cxx::project1st< _Arg1, _Arg2 >`
- struct `__gnu_cxx::project2nd< _Arg1, _Arg2 >`
- struct `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`
- class `__gnu_cxx::rope< _CharT, _Alloc >`
- struct `__gnu_cxx::select1st< _Pair >`
- struct `__gnu_cxx::select2nd< _Pair >`
- class `__gnu_cxx::slist< _Tp, _Alloc >`
- class `__gnu_cxx::subtractive_rng`
- struct `__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >`
- class `__gnu_cxx::unary_compose< _Operation1, _Operation2 >`

Functions

- `template<typename _Tp >`
`const _Tp & __gnu_cxx::__median (const _Tp &__a, const _Tp &__b, const _Tp &__c)`
- `template<typename _Tp, typename _Compare >`
`const _Tp & __gnu_cxx::__median (const _Tp &__a, const _Tp &__b, const _Tp &__c, _Compare __comp)`
- `size_t std::bitset< _Nb >::__Find_first () const noexcept`
- `size_t std::bitset< _Nb >::__Find_next (size_t __prev) const noexcept`
- `template<class _Operation1, class _Operation2 >`
`unary_compose< _Operation1, _Operation2 > __gnu_cxx::compose1 (const _Operation1 &__fn1, const _↔`
`Operation2 &__fn2)`

- `template<class _Operation1 , class _Operation2 , class _Operation3 >`
`binary_compose< _Operation1, _Operation2, _Operation3 > __gnu_cxx::compose2 (const _Operation1 &__fn1,`
`const _Operation2 &__fn2, const _Operation3 &__fn3)`
- `template<class _Result >`
`constant_void_fun< _Result > __gnu_cxx::constant0 (const _Result &__val)`
- `template<class _Result >`
`constant_unary_fun< _Result, _Result > __gnu_cxx::constant1 (const _Result &__val)`
- `template<class _Result >`
`constant_binary_fun< _Result, _Result, _Result > __gnu_cxx::constant2 (const _Result &__val)`
- `template<typename _InputIterator , typename _Size , typename _OutputIterator >`
`std::pair< _InputIterator, _OutputIterator > __gnu_cxx::copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`
- `template<typename _InputIterator , typename _Distance >`
`void __gnu_cxx::distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`
- `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::multiplies< _Tp >)`
- `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::plus< _Tp >)`
- `template<typename _InputIterator1 , typename _InputIterator2 >`
`int __gnu_cxx::lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _Tp , typename _Integer >`
`_Tp __gnu_cxx::power (_Tp __x, _Integer __n)`
- `template<typename _Tp , typename _Integer , typename _MonoidOperation >`
`_Tp __gnu_cxx::power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _InputIterator , typename _RandomAccessIterator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
- `template<typename _InputIterator , typename _RandomAccessIterator , typename _RandomNumberGenerator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator &__rand)`
- `template<typename _ForwardIterator , typename _OutputIterator , typename _Distance >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
- `template<typename _ForwardIterator , typename _OutputIterator , typename _Distance , typename _RandomNumberGenerator >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator &__rand)`
- `template<typename _InputIter , typename _Size , typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __result)`
- `bitset< _Nb > & std::bitset< _Nb >::Unchecked_set (size_t __pos) noexcept`
- `bitset< _Nb > & std::bitset< _Nb >::Unchecked_set (size_t __pos, int __val) noexcept`
- `bitset< _Nb > & std::bitset< _Nb >::Unchecked_reset (size_t __pos) noexcept`
- `bitset< _Nb > & std::bitset< _Nb >::Unchecked_flip (size_t __pos) noexcept`
- `constexpr bool std::bitset< _Nb >::Unchecked_test (size_t __pos) const noexcept`

3.6.4.1 Detailed Description

Because libstdc++ based its implementation of the STL subsections of the library on the SGI 3.3 implementation, we inherited their extensions as well.

They are additionally documented in the [online documentation](#), a copy of which is also shipped with the library source code (in `.../docs/html/documentation.html`). You can also read the documentation [on SGI's site](#), which is still running even though the code is not maintained.

NB that the following notes are pulled from various comments all over the place, so they may seem stilted.

The `identity_element` functions are not part of the C++ standard; SGI provided them as an extension. Its argument is an operation, and its return value is the identity element for that operation. It is overloaded for addition and multiplication, and you can overload it for your own nefarious operations.

As an extension to the binders, SGI provided composition functors and wrapper functions to aid in their creation. The `unary_compose` functor is constructed from two functions/functors, `f` and `g`. Calling `operator()` with a single argument `x` returns `f(g(x))`. The function `compose1` takes the two functions and constructs a `unary_compose` variable for you.

`binary_compose` is constructed from three functors, `f`, `g1`, and `g2`. Its `operator()` returns `f(g1(x),g2(x))`. The function `compose2` takes `f`, `g1`, and `g2`, and constructs the `binary_compose` instance for you. For example, if `f` returns an `int`, then

```
int answer = (compose2(f,g1,g2))(x);
```

is equivalent to

```
int temp1 = g1(x);
int temp2 = g2(x);
int answer = f(temp1,temp2);
```

But the first form is more compact, and can be passed around as a functor to other algorithms.

As an extension, SGI provided a functor called `identity`. When a functor is required but no operations are desired, this can be used as a pass-through. Its `operator()` returns its argument unchanged.

`select1st` and `select2nd` are extensions provided by SGI. Their `operator()`s take a `std::pair` as an argument, and return either the first member or the second member, respectively. They can be used (especially with the composition functors) to *strip* data from a sequence before performing the remainder of an algorithm.

The `operator()` of the `project1st` functor takes two arbitrary arguments and returns the first one, while `project2nd` returns the second one. They are extensions provided by SGI.

These three functors are each constructed from a single arbitrary variable/value. Later, their `operator()`s completely ignore any arguments passed, and return the stored value.

- `constant_void_fun`'s `operator()` takes no arguments
- `constant_unary_fun`'s `operator()` takes one argument (ignored)
- `constant_binary_fun`'s `operator()` takes two arguments (ignored)

The helper creator functions `constant0`, `constant1`, and `constant2` each take a *result* argument and construct variables of the appropriate functor type.

3.6.4.2 Function Documentation

`__median()` [1/2]

```
template<typename _Tp>
const _Tp & __gnu_cxx::__median (
    const _Tp & __a,
    const _Tp & __b,
    const _Tp & __c )
```

Find the median of three values.

Parameters

<code>__a</code>	A value.
<code>__b</code>	A value.
<code>__c</code>	A value.

Returns

One of a, b or c.

If {1,m,n} is some convolution of {a,b,c} such that $1 \leq m \leq n$ then the value returned will be m. This is an SGI extension.

__median() [2/2]

```
template<typename _Tp, typename _Compare >
const _Tp & __gnu_cxx::__median (
    const _Tp & __a,
    const _Tp & __b,
    const _Tp & __c,
    _Compare __comp )
```

Find the median of three values using a predicate for comparison.

Parameters

<code>__a</code>	A value.
<code>__b</code>	A value.
<code>__c</code>	A value.
<code>__comp</code>	A binary predicate.

Returns

One of a, b or c.

If {1,m,n} is some convolution of {a,b,c} such that `comp(1, m)` and `comp(m, n)` are both true then the value returned will be m. This is an SGI extension.

_Find_first()

```
template<size_t _Nb>
size_t std::bitset<_Nb>::_Find_first ( ) const [inline], [noexcept]
```

Finds the index of the first "on" bit.

Returns

The index of the first bit set, or `size()` if not found.

See also

[_Find_next](#)

_Find_next()

```
template<size_t _Nb>
size_t std::bitset<_Nb>::_Find_next (
    size_t __prev ) const [inline], [noexcept]
```

Finds the index of the next "on" bit after prev.

Returns

The index of the next bit set, or `size()` if not found.

Parameters

<code>__prev</code>	Where to start searching.
---------------------	---------------------------

See also

`_Find_first`**`_Unchecked_flip()`**

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::_Unchecked_flip (
    size_t __pos ) [inline], [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

`_Unchecked_reset()`

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::_Unchecked_reset (
    size_t __pos ) [inline], [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

`_Unchecked_set()` [1/2]

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::_Unchecked_set (
    size_t __pos ) [inline], [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

`_Unchecked_set()` [2/2]

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::_Unchecked_set (
    size_t __pos,
    int __val ) [inline], [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

`_Unchecked_test()`

```
template<size_t _Nb>
constexpr bool std::bitset< _Nb >::_Unchecked_test (
    size_t __pos ) const [inline], [constexpr], [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

`compose1()`

```
template<class _Operation1 , class _Operation2 >
unary_compose< _Operation1, _Operation2 > __gnu_cxx::compose1 (
    const _Operation1 & __fn1,
    const _Operation2 & __fn2 ) [inline]
```

An [SGI extension](#) .

compose2()

```
template<class _Operation1 , class _Operation2 , class _Operation3 >
binary_compose< _Operation1, _Operation2, _Operation3 > __gnu_cxx::compose2 (
    const _Operation1 & __fn1,
    const _Operation2 & __fn2,
    const _Operation3 & __fn3 ) [inline]
```

An [SGI extension](#) .

constant0()

```
template<class _Result >
constant_void_fun< _Result > __gnu_cxx::constant0 (
    const _Result & __val ) [inline]
```

An [SGI extension](#) .

constant1()

```
template<class _Result >
constant_unary_fun< _Result, _Result > __gnu_cxx::constant1 (
    const _Result & __val ) [inline]
```

An [SGI extension](#) .

constant2()

```
template<class _Result >
constant_binary_fun< _Result, _Result, _Result > __gnu_cxx::constant2 (
    const _Result & __val ) [inline]
```

An [SGI extension](#) .

copy_n()

```
template<typename _InputIterator , typename _Size , typename _OutputIterator >
std::pair< _InputIterator, _OutputIterator > __gnu_cxx::copy_n (
    _InputIterator __first,
    _Size __count,
    _OutputIterator __result ) [inline]
```

Copies the range [first,first+count) into [result,result+count).

Parameters

<code>__first</code>	An input iterator.
<code>__count</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

A std::pair composed of first+count and result+count.

This is an SGI extension. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

distance()

```
template<typename _InputIterator , typename _Distance >
```

```
void __gnu_cxx::distance (
    _InputIterator __first,
    _InputIterator __last,
    _Distance & __n ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

identity_element() [1/2]

```
template<class _Tp >
_Tp __gnu_cxx::identity_element (
    std::multiplies< _Tp > ) [inline]
```

An SGI extension .

identity_element() [2/2]

```
template<class _Tp >
_Tp __gnu_cxx::identity_element (
    std::plus< _Tp > ) [inline]
```

An SGI extension .

lexicographical_compare_3way()

```
template<typename _InputIterator1 , typename _InputIterator2 >
int __gnu_cxx::lexicographical_compare_3way (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2 )
```

memcmp on steroids.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

An int, as with memcmp.

The return value will be less than zero if the first range is *lexigraphically less than* the second, greater than zero if the second range is *lexigraphically less than* the first, and zero otherwise. This is an SGI extension.

power() [1/2]

```
template<typename _Tp , typename _Integer >
_Tp __gnu_cxx::power (
    _Tp __x,
    _Integer __n ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

power() [2/2]

```
template<typename _Tp , typename _Integer , typename _MonoidOperation >
_Tp __gnu_cxx::power (
    _Tp __x,
    _Integer __n,
    _MonoidOperation __monoid_op ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

random_sample() [1/2]

```
template<typename _InputIterator , typename _RandomAccessIterator >
_RandomAccessIterator __gnu_cxx::random_sample (
    _InputIterator __first,
    _InputIterator __last,
    _RandomAccessIterator __out_first,
    _RandomAccessIterator __out_last ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

random_sample() [2/2]

```
template<typename _InputIterator , typename _RandomAccessIterator , typename _RandomNumber↵
Generator >
_RandomAccessIterator __gnu_cxx::random_sample (
    _InputIterator __first,
    _InputIterator __last,
    _RandomAccessIterator __out_first,
    _RandomAccessIterator __out_last,
    _RandomNumberGenerator & __rand ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

random_sample_n() [1/2]

```
template<typename _ForwardIterator , typename _OutputIterator , typename _Distance >
_OutputIterator __gnu_cxx::random_sample_n (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _OutputIterator __out,
    const _Distance __n )
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

random_sample_n() [2/2]

```
template<typename _ForwardIterator , typename _OutputIterator , typename _Distance , typename _↵
RandomNumberGenerator >
_OutputIterator __gnu_cxx::random_sample_n (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _OutputIterator __out,
    const _Distance __n,
    _RandomNumberGenerator & __rand )
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

uninitialized_copy_n()

```
template<typename _InputIter , typename _Size , typename _ForwardIter >
std::pair< _InputIter, _ForwardIter > __gnu_cxx::uninitialized_copy_n (
    _InputIter __first,
    _Size __count,
    _ForwardIter __result ) [inline]
```

Copies the range [first,last) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__count</code>	Length
<code>__result</code>	An output iterator.

Returns

`__result + (__first + __count)`

Like `copy()`, but does not require an initialized output range.

3.7 File System

Files

- file [filesystem](#)

Classes

- struct [std::filesystem::__directory_iterator_proxy](#)
- class [std::filesystem::directory_entry](#)
- class [std::filesystem::directory_iterator](#)
- class [std::filesystem::file_status](#)
- class [std::filesystem::filesystem_error](#)
- class [std::filesystem::path::iterator](#)
- class [std::filesystem::path](#)
- class [std::filesystem::recursive_directory_iterator](#)
- struct [std::filesystem::space_info](#)

Typedefs

- using `std::filesystem::file_time_type` = `__file_clock::time_point`

Enumerations

- enum class `std::filesystem::copy_options` : unsigned short { `none` , `skip_existing` , `overwrite_existing` , `update_existing` , `recursive` , `copy_symlinks` , `skip_symlinks` , `directories_only` , `create_symlinks` , `create_hard_links` }
- enum class `std::filesystem::directory_options` : unsigned char { `none` , `follow_directory_symlink` , `skip_permission_denied` }
- enum class `std::filesystem::file_type` : signed char { `none` , `not_found` , `regular` , `directory` , `symlink` , `block` , `character` , `fifo` , `socket` , `unknown` }
- enum class `std::filesystem::perm_options` : unsigned { `replace` , `add` , `remove` , `nofollow` }
- enum class `std::filesystem::perms` : unsigned { `none` , `owner_read` , `owner_write` , `owner_exec` , `owner_all` , `group_read` , `group_write` , `group_exec` , `group_all` , `others_read` , `others_write` , `others_exec` , `others_all` , `all` , `set_uid` , `set_gid` , `sticky_bit` , `mask` , `unknown` }

Functions

- `path std::filesystem::absolute` (const `path` &__p)
- `path std::filesystem::absolute` (const `path` &__p, `error_code` &__ec)
- `path & std::filesystem::path::assign` (`string_type` &&__source)
- `iterator std::filesystem::path::begin` () const noexcept
- `path std::filesystem::canonical` (const `path` &__p)
- `path std::filesystem::canonical` (const `path` &__p, `error_code` &__ec)
- `int std::filesystem::path::compare` (const `string_type` &__s) const noexcept
- `int std::filesystem::path::compare` (const `value_type` * __s) const noexcept
- `void std::filesystem::copy` (const `path` &__from, const `path` &__to)
- `void std::filesystem::copy` (const `path` &__from, const `path` &__to, `copy_options` __options)
- `void std::filesystem::copy` (const `path` &__from, const `path` &__to, `copy_options` __options, `error_code` &__ec)
- `void std::filesystem::copy` (const `path` &__from, const `path` &__to, `error_code` &__ec)
- `bool std::filesystem::copy_file` (const `path` &__from, const `path` &__to)
- `bool std::filesystem::copy_file` (const `path` &__from, const `path` &__to, `copy_options` __option)
- `bool std::filesystem::copy_file` (const `path` &__from, const `path` &__to, `copy_options` __option, `error_code` &__ec)
- `bool std::filesystem::copy_file` (const `path` &__from, const `path` &__to, `error_code` &__ec)
- `void std::filesystem::copy_symlink` (const `path` &__existing_symlink, const `path` &__new_symlink)
- `void std::filesystem::copy_symlink` (const `path` &__existing_symlink, const `path` &__new_symlink, `error_code` &__ec) noexcept
- `bool std::filesystem::create_directories` (const `path` &__p)
- `bool std::filesystem::create_directories` (const `path` &__p, `error_code` &__ec)
- `bool std::filesystem::create_directory` (const `path` &__p)
- `bool std::filesystem::create_directory` (const `path` &__p, const `path` &__attributes)
- `bool std::filesystem::create_directory` (const `path` &__p, const `path` &__attributes, `error_code` &__ec) noexcept
- `bool std::filesystem::create_directory` (const `path` &__p, `error_code` &__ec) noexcept
- `void std::filesystem::create_directory_symlink` (const `path` &__to, const `path` &__new_symlink)

- void **std::filesystem::create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- void **std::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link)
- void **std::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link, [error_code](#) &__ec) noexcept
- void **std::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **std::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- [path](#) **std::filesystem::current_path** ()
- void **std::filesystem::current_path** (const [path](#) &__p)
- void **std::filesystem::current_path** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) **std::filesystem::current_path** ([error_code](#) &__ec)
- [iterator](#) **std::filesystem::path::end** () const noexcept
- bool **std::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2)
- bool **std::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2, [error_code](#) &__ec) noexcept
- bool **std::filesystem::exists** (const [path](#) &__p)
- bool **std::filesystem::exists** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::exists** ([file_status](#)) noexcept
- [path](#) **std::filesystem::path::extension** () const
- [uintmax_t](#) **std::filesystem::file_size** (const [path](#) &)
- [uintmax_t](#) **std::filesystem::file_size** (const [path](#) &, [error_code](#) &) noexcept
- [path](#) **std::filesystem::path::filename** () const
- [std::string](#) **std::filesystem::path::generic_string** () const
- [template](#)<typename [_CharT](#), typename [_Traits](#) = [std::char_traits](#)<[_CharT](#)>, typename [_Allocator](#) = [std::allocator](#)<[_CharT](#)>>> [std::basic_string](#)<[_CharT](#), [_Traits](#), [_Allocator](#)> **std::filesystem::path::generic_string** (const [_Allocator](#) &__a= [_Allocator](#)()) const
- [std::u16string](#) **std::filesystem::path::generic_u16string** () const
- [std::u32string](#) **std::filesystem::path::generic_u32string** () const
- [std::string](#) **std::filesystem::path::generic_u8string** () const
- [std::wstring](#) **std::filesystem::path::generic_wstring** () const
- [uintmax_t](#) **std::filesystem::hard_link_count** (const [path](#) &)
- [uintmax_t](#) **std::filesystem::hard_link_count** (const [path](#) &, [error_code](#) &) noexcept
- bool **std::filesystem::path::has_extension** () const noexcept
- bool **std::filesystem::path::has_stem** () const noexcept
- [size_t](#) **std::filesystem::hash_value** (const [path](#) &__p) noexcept
- bool **std::filesystem::path::is_absolute** () const noexcept
- bool **std::filesystem::is_block_file** (const [path](#) &__p)
- bool **std::filesystem::is_block_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_block_file** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_character_file** (const [path](#) &__p)
- bool **std::filesystem::is_character_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_character_file** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_directory** (const [path](#) &__p)
- bool **std::filesystem::is_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_directory** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_empty** (const [path](#) &__p)
- bool **std::filesystem::is_empty** (const [path](#) &__p, [error_code](#) &__ec)
- bool **std::filesystem::is_fifo** (const [path](#) &__p)
- bool **std::filesystem::is_fifo** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_fifo** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_other** (const [path](#) &__p)

- `bool std::filesystem::is_other (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_other (file_status) noexcept`
- `bool std::filesystem::is_regular_file (const path &__p)`
- `bool std::filesystem::is_regular_file (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_regular_file (file_status) noexcept`
- `bool std::filesystem::is_socket (const path &__p)`
- `bool std::filesystem::is_socket (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_socket (file_status __s) noexcept`
- `bool std::filesystem::is_symlink (const path &__p)`
- `bool std::filesystem::is_symlink (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_symlink (file_status) noexcept`
- `file_time_type std::filesystem::last_write_time (const path &)`
- `file_time_type std::filesystem::last_write_time (const path &, error_code &) noexcept`
- `void std::filesystem::last_write_time (const path &__p, file_time_type __new_time)`
- `void std::filesystem::last_write_time (const path &__p, file_time_type __new_time, error_code &__ec) noexcept`
- `path & std::filesystem::path::make_preferred ()`
- `copy_options & std::filesystem::operator+= (copy_options &__x, copy_options __y) noexcept`
- `reference std::filesystem::path::iterator::operator* () const noexcept`
- `iterator & std::filesystem::path::iterator::operator++ () noexcept`
- `template<typename _CharT >
__detail::_Path2< _CharT * > & std::filesystem::path::operator+= (_CharT __x)`
- `path & std::filesystem::path::operator+= (basic_string_view< value_type > __x)`
- `path & std::filesystem::path::operator+= (const string_type &__x)`
- `path & std::filesystem::path::operator+= (const value_type * __x)`
- `path & std::filesystem::path::operator+= (value_type __x)`
- `iterator & std::filesystem::path::iterator::operator-- () noexcept`
- `path & std::filesystem::path::operator= (path &&) noexcept`
- `path & std::filesystem::path::operator= (string_type && __source)`
- `constexpr copy_options std::filesystem::operator^ (copy_options __x, copy_options __y) noexcept`
- `copy_options & std::filesystem::operator^= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options std::filesystem::operator| (copy_options __x, copy_options __y) noexcept`
- `copy_options & std::filesystem::operator|= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options std::filesystem::operator~ (copy_options __x) noexcept`
- `void std::filesystem::permissions (const path &, perms, perm_options, error_code &) noexcept`
- `void std::filesystem::permissions (const path &__p, perms __prms, error_code &__ec) noexcept`
- `void std::filesystem::permissions (const path &__p, perms __prms, perm_options __opts=perm_options::replace)`
- `path std::filesystem::proximate (const path &__p, const path &__base, error_code &__ec)`
- `path std::filesystem::proximate (const path &__p, const path &__base=current_path())`
- `path std::filesystem::proximate (const path &__p, error_code &__ec)`
- `path std::filesystem::read_symlink (const path &__p)`
- `path std::filesystem::read_symlink (const path &__p, error_code &__ec)`
- `path std::filesystem::relative (const path &__p, const path &__base, error_code &__ec)`
- `path std::filesystem::relative (const path &__p, const path &__base=current_path())`
- `path std::filesystem::relative (const path &__p, error_code &__ec)`
- `bool std::filesystem::remove (const path &, error_code &) noexcept`
- `bool std::filesystem::remove (const path &__p)`
- `uintmax_t std::filesystem::remove_all (const path &)`
- `uintmax_t std::filesystem::remove_all (const path &, error_code &)`
- `void std::filesystem::rename (const path &__from, const path &__to)`

- void **std::filesystem::rename** (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec) noexcept
- void **std::filesystem::resize_file** (const [path](#) &__p, uintmax_t __size)
- void **std::filesystem::resize_file** (const [path](#) &__p, uintmax_t __size, [error_code](#) &__ec) noexcept
- [space_info](#) **std::filesystem::space** (const [path](#) &__p)
- [space_info](#) **std::filesystem::space** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [file_status](#) **std::filesystem::status** (const [path](#) &__p)
- [file_status](#) **std::filesystem::status** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::status_known** ([file_status](#)) noexcept
- [path](#) **std::filesystem::path::stem** () const
- [string](#) **std::filesystem::path::string** () const
- template<typename _CharT, typename _Traits, typename _Allocator>
[basic_string](#)<_CharT, _Traits, _Allocator> **std::filesystem::path::string** (const _Allocator &__a) const
- void **std::filesystem::path::swap** ([path](#) &__rhs) noexcept
- void **swap** (same_as<[path](#)> auto &__lhs, same_as<[path](#)> auto &__rhs) noexcept
- [file_status](#) **std::filesystem::symlink_status** (const [path](#) &__p)
- [file_status](#) **std::filesystem::symlink_status** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) **std::filesystem::temp_directory_path** ()
- [path](#) **std::filesystem::temp_directory_path** ([error_code](#) &__ec)
- [u16string](#) **std::filesystem::path::u16string** () const
- [u32string](#) **std::filesystem::path::u32string** () const
- template<typename _InputIterator, typename _Require = __detail::_Path2<_InputIterator>, typename _CharT = __detail::_value_type<__is_char_or_char8_t<_InputIterator>>>
[path](#) **u8path** (_InputIterator __first, _InputIterator __last)
- template<typename _Source, typename _Require = __detail::_Path<_Source>, typename _CharT = __detail::_value_type_is_char_or__char8_t<_Source>>
[path](#) **u8path** (const _Source &__source)
- [string](#) **std::filesystem::path::u8string** () const
- [path](#) **std::filesystem::weakly_canonical** (const [path](#) &__p)
- [path](#) **std::filesystem::weakly_canonical** (const [path](#) &__p, [error_code](#) &__ec)
- [wstring](#) **std::filesystem::path::wstring** () const
- constexpr [perms](#) **std::filesystem::operator|** ([perms](#) __x, [perms](#) __y) noexcept
- constexpr [perms](#) **std::filesystem::operator^** ([perms](#) __x, [perms](#) __y) noexcept
- constexpr [perms](#) **std::filesystem::operator~** ([perms](#) __x) noexcept
- [perms](#) & **std::filesystem::operator&=** ([perms](#) &__x, [perms](#) __y) noexcept
- [perms](#) & **std::filesystem::operator|=** ([perms](#) &__x, [perms](#) __y) noexcept
- [perms](#) & **std::filesystem::operator^=** ([perms](#) &__x, [perms](#) __y) noexcept
- constexpr [perm_options](#) **std::filesystem::operator|** ([perm_options](#) __x, [perm_options](#) __y) noexcept
- constexpr [perm_options](#) **std::filesystem::operator^** ([perm_options](#) __x, [perm_options](#) __y) noexcept
- constexpr [perm_options](#) **std::filesystem::operator~** ([perm_options](#) __x) noexcept
- [perm_options](#) & **std::filesystem::operator&=** ([perm_options](#) &__x, [perm_options](#) __y) noexcept
- [perm_options](#) & **std::filesystem::operator|=** ([perm_options](#) &__x, [perm_options](#) __y) noexcept
- [perm_options](#) & **std::filesystem::operator^=** ([perm_options](#) &__x, [perm_options](#) __y) noexcept
- constexpr [directory_options](#) **std::filesystem::operator|** ([directory_options](#) __x, [directory_options](#) __y) noexcept
- constexpr [directory_options](#) **std::filesystem::operator^** ([directory_options](#) __x, [directory_options](#) __y) noexcept
- constexpr [directory_options](#) **std::filesystem::operator~** ([directory_options](#) __x) noexcept
- [directory_options](#) & **std::filesystem::operator&=** ([directory_options](#) &__x, [directory_options](#) __y) noexcept
- [directory_options](#) & **std::filesystem::operator|=** ([directory_options](#) &__x, [directory_options](#) __y) noexcept
- [directory_options](#) & **std::filesystem::operator^=** ([directory_options](#) &__x, [directory_options](#) __y) noexcept
- [directory_iterator](#) **begin** ([directory_iterator](#) __iter) noexcept
- [directory_iterator](#) **end** ([directory_iterator](#)) noexcept
- [recursive_directory_iterator](#) **begin** ([recursive_directory_iterator](#) __iter) noexcept
- [recursive_directory_iterator](#) **end** ([recursive_directory_iterator](#)) noexcept

3.7.1 Detailed Description

Utilities for performing operations on file systems and their components, such as paths, regular files, and directories.

3.7.2 Typedef Documentation

file_time_type

```
using std::filesystem::file_time_type = typedef __file_clock::time_point
```

The type used for file timestamps.

3.7.3 Enumeration Type Documentation

copy_options

```
enum class std::filesystem::copy_options : unsigned short [strong]
```

Bitmask type controlling effects of `filesystem::copy`

directory_options

```
enum class std::filesystem::directory_options : unsigned char [strong]
```

Bitmask type controlling directory iteration.

file_type

```
enum class std::filesystem::file_type : signed char [strong]
```

Enumerated type representing the type of a file.

perm_options

```
enum class std::filesystem::perm_options : unsigned [strong]
```

Bitmask type controlling changes to permissions.

perms

```
enum class std::filesystem::perms : unsigned [strong]
```

Bitmask type representing file access permissions.

3.7.4 Function Documentation

begin() [1/2]

```
directory_iterator begin (  
    directory_iterator __iter ) [related]
```

Enable range-based for using `directory_iterator`.

e.g. for (auto& entry : `std::filesystem::directory_iterator`(".")) ...

begin() [2/2]

```
recursive_directory_iterator begin (  
    recursive_directory_iterator __iter ) [related]
```

Enable range-based for using `recursive_directory_iterator`.

e.g. for (auto& entry : `recursive_directory_iterator`(".")) ...

end() [1/2]

```
directory_iterator end (
    directory_iterator ) [related]
```

Return a past-the-end `directory_iterator`.

end() [2/2]

```
recursive_directory_iterator end (
    recursive_directory_iterator ) [related]
```

Return a past-the-end `recursive_directory_iterator`.

u8path() [1/2]

```
template<typename _InputIterator , typename _Require = __detail::_Path2<_InputIterator>, typename
_CharT = __detail::_value_type_is_char_or_char8_t<_InputIterator>>
```

```
path u8path (
    _InputIterator __first,
    _InputIterator __last ) [related]
```

Create a path from a UTF-8-encoded sequence of char

u8path() [2/2]

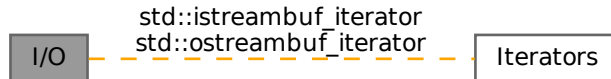
```
template<typename _Source , typename _Require = __detail::_Path<_Source>, typename _CharT = __↔
detail::_value_type_is_char_or_char8_t<_Source>>
```

```
path u8path (
    const _Source & __source ) [related]
```

Create a path from a UTF-8-encoded sequence of char

3.8 I/O

Collaboration diagram for I/O:

**Classes**

- class `std::basic_filebuf<_CharT, _Traits>`
- class `std::basic_fstream<_CharT, _Traits>`
- class `std::basic_ifstream<_CharT, _Traits>`
- class `std::basic_ios<_CharT, _Traits>`
- class `std::basic_iostream<_CharT, _Traits>`
- class `std::basic_istream<_CharT, _Traits>`
- class `std::basic_istreamstream<_CharT, _Traits, _Alloc>`
- class `std::basic_ofstream<_CharT, _Traits>`

- class `std::basic_ostream< _CharT, _Traits >`
- class `std::basic_ostringstream< _CharT, _Traits, _Alloc >`
- class `std::basic_streambuf< _CharT, _Traits >`
- class `std::basic_stringbuf< _CharT, _Traits, _Alloc >`
- class `std::basic_stringstream< _CharT, _Traits, _Alloc >`
- class `std::ios_base::failure`
- class `std::ios_base`
- class `std::istreambuf_iterator< _CharT, _Traits >`
- class `std::ostreambuf_iterator< _CharT, _Traits >`
- class `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`
- class `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`

Typedefs

- typedef `basic_filebuf< char > std::filebuf`
- typedef `basic_fstream< char > std::fstream`
- typedef `basic_ifstream< char > std::ifstream`
- typedef `basic_ios< char > std::ios`
- typedef `basic_iostream< char > std::iostream`
- typedef `basic_istream< char > std::istream`
- typedef `basic_istreamstream< char > std::istreamstream`
- typedef `basic_ofstream< char > std::ofstream`
- typedef `basic_ostream< char > std::ostream`
- typedef `basic_ostringstream< char > std::ostringstream`
- typedef `basic_streambuf< char > std::streambuf`
- typedef `basic_stringbuf< char > std::stringbuf`
- typedef `basic_stringstream< char > std::stringstream`
- typedef `basic_filebuf< wchar_t > std::wfilebuf`
- typedef `basic_fstream< wchar_t > std::wfstream`
- typedef `basic_ifstream< wchar_t > std::wifstream`
- typedef `basic_ios< wchar_t > std::wios`
- typedef `basic_iostream< wchar_t > std::wiostream`
- typedef `basic_istream< wchar_t > std::wistream`
- typedef `basic_istreamstream< wchar_t > std::wistreamstream`
- typedef `basic_ofstream< wchar_t > std::wofstream`
- typedef `basic_ostream< wchar_t > std::wostream`
- typedef `basic_ostringstream< wchar_t > std::wostringstream`
- typedef `basic_streambuf< wchar_t > std::wstreambuf`
- typedef `basic_stringbuf< wchar_t > std::wstringbuf`
- typedef `basic_stringstream< wchar_t > std::wstringstream`

3.8.1 Detailed Description

Nearly all of the I/O classes are parameterized on the type of characters they read and write. (The major exception is `ios_base` at the top of the hierarchy.) This is a change from pre-Standard streams, which were not templates.

For ease of use and compatibility, all of the `basic_*` I/O-related classes are given typedef names for both of the builtin character widths (wide and narrow). The typedefs are the same as the pre-Standard names, for example:

```
typedef basic_ifstream<char> ifstream;
```

Because properly forward-declaring these classes can be difficult, you should not do it yourself. Instead, include the `<iosfwd>` header, which contains only declarations of all the I/O classes as well as the typedefs. Trying to forward-declare the typedefs themselves (e.g., `class ostream;`) is not valid ISO C++.

For more specific declarations, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html#std.io.objects>

3.8.2 Typedef Documentation

filebuf

```
typedef basic_filebuf<char> std::filebuf
```

Class for `char` file buffers.

fstream

```
typedef basic_fstream<char> std::fstream
```

Class for `char` mixed input and output file streams.

ifstream

```
typedef basic_ifstream<char> std::ifstream
```

Class for `char` input file streams.

ios

```
typedef basic_ios<char> std::ios
```

Base class for `char` streams.

iostream

```
typedef basic_iostream<char> std::iostream
```

Base class for `char` mixed input and output streams.

istream

```
typedef basic_istream<char> std::istream
```

Base class for `char` input streams.

istringstream

```
typedef basic_istringstream<char> std::istringstream
```

Class for `char` input memory streams.

ofstream

```
typedef basic_ofstream<char> std::ofstream
```

Class for `char` output file streams.

ostream

```
typedef basic_ostream<char> std::ostream
```

Base class for `char` output streams.

ostringstream

```
typedef basic_ostringstream<char> std::ostringstream
```

Class for `char` output memory streams.

streambuf

```
typedef basic_streambuf<char> std::streambuf
```

Base class for `char` buffers.

stringbuf

typedef `basic_stringbuf<char>` `std::stringbuf`
Class for `char` memory buffers.

stringstream

typedef `basic_stringstream<char>` `std::stringstream`
Class for `char` mixed input and output memory streams.

wfilebuf

typedef `basic_filebuf<wchar_t>` `std::wfilebuf`
Class for `wchar_t` file buffers.

wfstream

typedef `basic_fstream<wchar_t>` `std::wfstream`
Class for `wchar_t` mixed input and output file streams.

wifstream

typedef `basic_ifstream<wchar_t>` `std::wifstream`
Class for `wchar_t` input file streams.

wios

typedef `basic_ios<wchar_t>` `std::wios`
Base class for `wchar_t` streams.

wiostream

typedef `basic_iostream<wchar_t>` `std::wiostream`
Base class for `wchar_t` mixed input and output streams.

wistream

typedef `basic_istream<wchar_t>` `std::wistream`
Base class for `wchar_t` input streams.

wistringstream

typedef `basic_istringstream<wchar_t>` `std::wistringstream`
Class for `wchar_t` input memory streams.

wofstream

typedef `basic_ofstream<wchar_t>` `std::wofstream`
Class for `wchar_t` output file streams.

wostream

typedef `basic_ostream<wchar_t>` `std::wostream`
Base class for `wchar_t` output streams.

wostreamstream

typedef `basic_ostringstream<wchar_t>` `std::wostreamstream`
 Class for `wchar_t` output memory streams.

wstreambuf

typedef `basic_streambuf<wchar_t>` `std::wstreambuf`
 Base class for `wchar_t` buffers.

wstringbuf

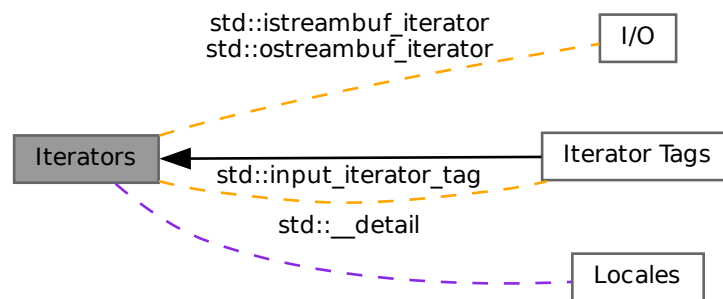
typedef `basic_stringbuf<wchar_t>` `std::wstringbuf`
 Class for `wchar_t` memory buffers.

wstringstream

typedef `basic_stringstream<wchar_t>` `std::wstringstream`
 Class for `wchar_t` mixed input and output memory streams.

3.9 Iterators

Collaboration diagram for Iterators:

**Modules**

- [Iterator Tags](#)

Namespaces

- namespace `std::_detail`

Classes

- class `std::back_insert_iterator<_Container>`
- class `std::common_iterator<_It, _Sent>`
- class `std::counted_iterator<_It>`
- class `std::front_insert_iterator<_Container>`

- struct `std::input_iterator_tag`
- class `std::insert_iterator< _Container >`
- class `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >`
- class `std::istreambuf_iterator< _CharT, _Traits >`
- struct `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`
- struct `std::iterator_traits< _Iterator >`
- struct `std::iterator_traits< _Tp * >`
- class `std::move_iterator< _Iterator >`
- class `std::ostream_iterator< _Tp, _CharT, _Traits >`
- class `std::ostreambuf_iterator< _CharT, _Traits >`
- class `std::reverse_iterator< _Iterator >`

Macros

- `#define __cpp_lib_make_reverse_iterator`

Functions

- template<bool _IsMove, typename _CharT >
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT >::__type std::__copy_move_a2 (_CharT * __first, _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- template<bool _IsMove, typename _CharT >
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT >::__type std::__copy_move_a2 (const _CharT * __first, const _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- template<bool _IsMove, typename _CharT >
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type std::__copy_move_a2 (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, _CharT * __result)`
- template<typename _CharT, typename _Size >
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type std::__copy_n_a (istreambuf_iterator< _CharT > __it, _Size __n, _CharT * __result, bool __strict)`
- template<typename _Iter >
`constexpr iterator_traits< _Iter >::iterator_category std::__iterator_category (const _Iter &)`
- template<typename _Iterator, typename _ReturnType = __conditional_t<__move_if_noexcept_cond<typename iterator_traits<_Iterator>::value_type>::value, _Iterator, move_iterator<_Iterator>>>
`constexpr _ReturnType std::__make_move_if_noexcept_iterator (_Iterator __i)`
- template<typename _Tp, typename _ReturnType = __conditional_t<__move_if_noexcept_cond<_Tp>::value, const _Tp*, move_iterator<_Tp*>>>
`constexpr _ReturnType std::__make_move_if_noexcept_iterator (_Tp * __i)`
- template<typename _Iterator >
`constexpr reverse_iterator< _Iterator > std::__make_reverse_iterator (_Iterator __i)`
- template<typename _Iterator >
`constexpr auto std::__niter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- template<typename _Iterator >
`constexpr auto std::__niter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- template<typename _CharT, typename _Distance >
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, void >::__type std::advance (istreambuf_iterator< _CharT > & __i, _Distance __n)`
- template<typename _Container >
`constexpr back_insert_iterator< _Container > std::back_inserter (_Container & __x)`

- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT >::__type std::copy`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, ostreambuf_iterator< _CharT >`
`__result)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT >::__type std::find`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT &__val)`
- `template<typename _Container >`
`constexpr front_insert_iterator< _Container > std::front_inserter (_Container &__x)`
- `template<typename _Container >`
`constexpr insert_iterator< _Container > std::inserter (_Container &__x, std::__detail::__range_iter_t< _Container > __i)`
- `template<typename _Iterator >`
`constexpr move_iterator< _Iterator > std::make_move_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`constexpr reverse_iterator< _Iterator > std::make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`constexpr bool std::operator!= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr bool std::operator!= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR >`
`&__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() != __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator!= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR`
`> &__y)`
- `template<typename _Iterator >`
`constexpr move_iterator< _Iterator > std::operator+ (typename move_iterator< _Iterator >::difference_type __n,`
`const move_iterator< _Iterator > &__x)`
- `template<typename _Iterator >`
`constexpr reverse_iterator< _Iterator > std::operator+ (typename reverse_iterator< _Iterator >::difference_type __n,`
`const reverse_iterator< _Iterator > &__x)`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto std::operator- (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y) -> decltype(__x.base() - __y.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto std::operator- (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR >`
`&__y) -> decltype(__y.base() - __x.base())`
- `template<typename _Iterator >`
`constexpr bool std::operator< (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator< (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR >`
`&__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() > __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator< (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR`
`> &__y)`
- `template<typename _Iterator >`
`constexpr bool std::operator<= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator<= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR >`
`&__y)`

- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() >= __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator<= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<three_way_comparable _Iterator>`
`constexpr compare_three_way_result_t< _Iterator, _Iterator > std::operator<=> (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, three_way_comparable_with< _IteratorL > _IteratorR>`
`constexpr compare_three_way_result_t< _IteratorL, _IteratorR > std::operator<=> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _CharT, typename _Traits >`
`bool std::operator== (const istreambuf_iterator< _CharT, _Traits > &__a, const istreambuf_iterator< _CharT, _Traits > &__b)`
- `template<typename _Iterator >`
`constexpr bool std::operator== (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`constexpr bool std::operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`constexpr bool std::operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator>= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() <= __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator>= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`

Variables

- `template<typename _Iterator1, typename _Iterator2 >`
`constexpr bool std::disable_sized_sentinel_for< reverse_iterator< _Iterator1 >, reverse_iterator< _Iterator2 > >`

3.9.1 Detailed Description

Abstractions for uniform iterating through various underlying types.

3.9.2 Function Documentation

`__iterator_category()`

```
template<typename _Iter >
constexpr iterator_traits< _Iter >::iterator_category std::__iterator_category (
    const _Iter & ) [inline], [constexpr]
```

This function is not a part of the C++ standard but is syntactic sugar for internal library use only.

Referenced by `std::deque<_Tp, _Alloc>::deque()`, `std::vector<_Tp, _Alloc>::vector()`, `std::__find_if_not()`, `__gnu_debug::__valid_range_aux()`, `std::advance()`, `std::deque<_Tp, _Alloc>::assign()`, `std::copy_n()`, `std::distance()`, `std::fill_n()`, `std::find_end()`, `std::find_end()`, `std::deque<_Tp, _Alloc>::insert()`, `std::partition()`, `std::reverse()`, `std::rotate()`, `std::uninitialized_copy_n()`, `std::unique_copy()`, and `std::unique_copy()`.

`back_inserter()`

```
template<typename _Container >
constexpr back_insert_iterator< _Container > std::back_inserter (
    _Container & __x ) [inline], [constexpr]
```

Parameters

<code>__x</code>	A container of arbitrary type.
------------------	--------------------------------

Returns

An instance of `back_insert_iterator` working on `__x`.

This wrapper function helps in creating `back_insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

Referenced by `std::match_results<_Bi_iter, _Alloc>::format()`, `std::match_results<_Bi_iter, _Alloc>::format()`, `std::regex_replace()`, `std::regex_replace()`, `std::regex_replace()`, and `std::regex_replace()`.

`front_inserter()`

```
template<typename _Container >
constexpr front_insert_iterator< _Container > std::front_inserter (
    _Container & __x ) [inline], [constexpr]
```

Parameters

<code>__x</code>	A container of arbitrary type.
------------------	--------------------------------

Returns

An instance of `front_insert_iterator` working on `x`.

This wrapper function helps in creating `front_insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

inserter()

```
template<typename _Container >
constexpr insert_iterator< _Container > std::inserter (
    _Container & __x,
    std::__detail::__range_iter_t< _Container > __i ) [constexpr]
```

Parameters

__x	A container of arbitrary type.
__i	An iterator into the container.

Returns

An instance of `insert_iterator` working on `__x`.

This wrapper function helps in creating `insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

make_reverse_iterator()

```
template<typename _Iterator >
constexpr reverse_iterator< _Iterator > std::make_reverse_iterator (
    _Iterator __i ) [inline], [constexpr]
```

Generator function for `reverse_iterator`.

operator==()

```
template<typename _IteratorL , typename _IteratorR >
requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }
constexpr bool std::operator== (
    const reverse_iterator< _IteratorL > & __x,
    const reverse_iterator< _IteratorR > & __y ) [constexpr]
```

Parameters

__x	A <code>reverse_iterator</code> .
__y	A <code>reverse_iterator</code> .

Returns

A simple bool.

Reverse iterators forward comparisons to their underlying base() iterators.

References [std::reverse_iterator<_Iterator>::base\(\)](#).

3.9.3 Iterator Tags

Collaboration diagram for Iterator Tags:

**Classes**

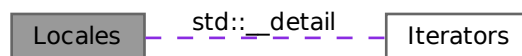
- struct [std::bidirectional_iterator_tag](#)
- struct [std::contiguous_iterator_tag](#)
- struct [std::forward_iterator_tag](#)
- struct [std::input_iterator_tag](#)
- struct [std::output_iterator_tag](#)
- struct [std::random_access_iterator_tag](#)

3.9.3.1 Detailed Description

These are empty types, used to distinguish different iterators. The distinction is not made by what they contain, but simply by what they are. Different underlying algorithms can then be used based on the different operations supported by different iterator types.

3.10 Locales

Collaboration diagram for Locales:

**Namespaces**

- namespace [std::__detail](#)

Classes

- class `std::codecvt< _InternT, _ExternT, _StateT >`
- class `std::ctype< _CharT >`
- class `std::ctype< char >`
- class `std::ctype< wchar_t >`
- class `std::locale::facet`
- class `std::locale::id`
- class `std::locale`
- class `std::messages< _CharT >`
- struct `std::messages_base`
- class `std::money_base`
- class `std::money_get< _CharT, _InIter >`
- class `std::money_put< _CharT, _OutIter >`
- class `std::moneypunct< _CharT, _Intl >`
- class `std::num_get< _CharT, _InIter >`
- class `std::num_put< _CharT, _OutIter >`
- class `std::numpunct< _CharT >`
- class `std::time_base`
- class `std::time_get< _CharT, _InIter >`
- class `std::time_put< _CharT, _OutIter >`
- class `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`
- class `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >`

Functions

- template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >
bool **std::__do_str_codecvt** (const _InChar *__first, const _InChar *__last, _OutStr &__outstr, const _Codecvt &__cvt, _State &__state, size_t &__count, _Fn __fn)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_in** (const char *__first, const char *__last, [basic_string](#)< _CharT, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_in** (const char *__first, const char *__last, [basic_string](#)< _CharT, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_in_all** (const char *__first, const char *__last, [basic_string](#)< _CharT, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_out** (const _CharT *__first, const _CharT *__last, [basic_string](#)< char, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_out** (const _CharT *__first, const _CharT *__last, [basic_string](#)< char, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)
- template<typename _CharT, typename _Traits, typename _Alloc, typename _State >
bool **std::__str_codecvt_out_all** (const _CharT *__first, const _CharT *__last, [basic_string](#)< char, _Traits, _Alloc > &__outstr, const [codecvt](#)< _CharT, char, _State > &__cvt)
- template<typename _Facet >
bool **std::has_facet** (const [locale](#) &__loc) throw ()
- template<typename _Facet >
const _Facet & **std::use_facet** (const [locale](#) &__loc)

3.10.1 Detailed Description

Classes and functions for internationalization and localization.

3.10.2 Function Documentation

has_facet()

```
template<typename _Facet >
bool std::has_facet (
    const locale & __loc ) throw ( )
```

Test for the presence of a facet.

has_facet tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

use_facet()

```
template<typename _Facet >
const _Facet & std::use_facet (
    const locale & __loc )
```

Return a facet.

use_facet looks for and returns a reference to a facet of type Facet where Facet is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type Facet.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

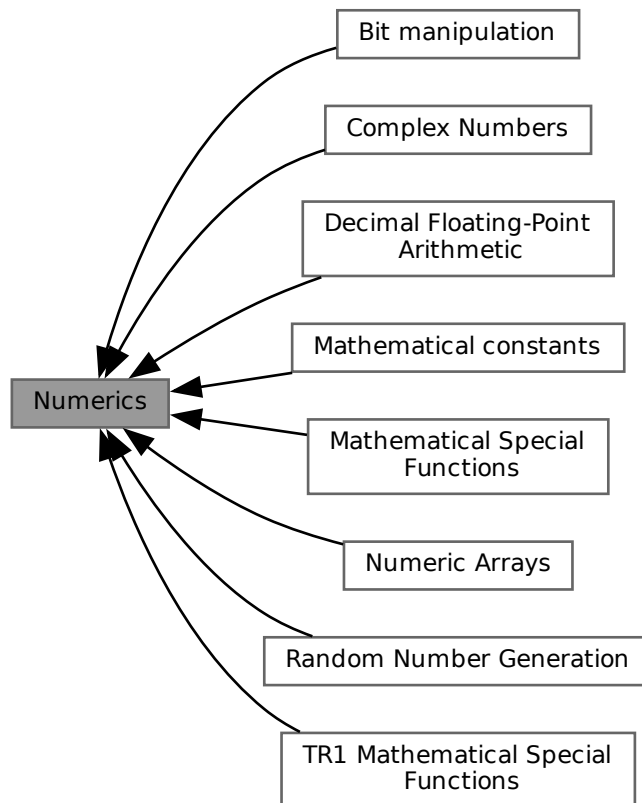
Reference to facet of type Facet.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

3.11 Numerics

Collaboration diagram for Numerics:



Modules

- [Bit manipulation](#)
- [Complex Numbers](#)
- [Decimal Floating-Point Arithmetic](#)
- [Mathematical Special Functions](#)
- [Mathematical constants](#)
- [Numeric Arrays](#)
- [Random Number Generation](#)
- [TR1 Mathematical Special Functions](#)

3.11.1 Detailed Description

Components for performing numeric operations. Includes support for complex number types, random number generation, numeric (n-at-a-time) arrays, generalized numeric algorithms, and mathematical special functions.

3.11.2 Bit manipulation

Collaboration diagram for Bit manipulation:



Macros

- `#define __cpp_lib_bit_cast`
- `#define __cpp_lib_bitops`
- `#define __cpp_lib_endian`
- `#define __cpp_lib_int_pow2`

Enumerations

- enum class `std::endian` { `little`, `big`, `native` }

Functions

- `template<typename _To, typename _From >`
`requires (sizeof(_To) == sizeof(_From)) && __is_trivially_copyable(_To) && __is_trivially_copyable(_From)`
`constexpr _To std::bit_cast (const _From &__from) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > std::bit_ceil (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > std::bit_floor (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > std::bit_width (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > std::countl_one (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > std::countl_zero (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > std::countr_one (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > std::countr_zero (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, bool > std::has_single_bit (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > std::popcount (_Tp __x) noexcept`

- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > std::rotl (_Tp __x, int __s) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > std::rotr (_Tp __x, int __s) noexcept`

3.11.2.1 Detailed Description

Utilities for examining and manipulating individual bits.

3.11.2.2 Enumeration Type Documentation

endian

```
enum class std::endian [strong]
```

Byte order constants.

The platform endianness can be checked by comparing `std::endian::native` to one of `std::endian::big` or `std::endian::little`.

Since

C++20

3.11.2.3 Function Documentation

bit_cast()

```
template<typename _To , typename _From >  
requires (sizeof(_To) == sizeof(_From)) && __is_trivially_copyable(_To) && __is_trivially_copyable(←  
_From)  
constexpr _To std::bit_cast (   
    const _From & __from ) [constexpr], [noexcept]
```

Create a value of type `To` from the bits of `from`.

Template Parameters

<code>_To</code>	A trivially-copyable type.
------------------	----------------------------

Parameters

<code>__from</code>	A trivially-copyable object of the same size as <code>_To</code> .
---------------------	--

Returns

An object of type `_To`.

Since

C++20

bit_ceil()

```
template<typename _Tp >  
constexpr _If_is_unsigned_integer< _Tp > std::bit_ceil (   
    _Tp __x ) [constexpr], [noexcept]
```

The smallest power-of-two not less than `x`.

bit_floor()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp > std::bit_floor (
    _Tp __x ) [constexpr], [noexcept]
```

The largest power-of-two not greater than *x*.

bit_width()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp > std::bit_width (
    _Tp __x ) [constexpr], [noexcept]
```

The smallest integer greater than the base-2 logarithm of *x*.

countl_one()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > std::countl_one (
    _Tp __x ) [constexpr], [noexcept]
```

The number of contiguous one bits, starting from the highest bit.

countl_zero()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > std::countl_zero (
    _Tp __x ) [constexpr], [noexcept]
```

The number of contiguous zero bits, starting from the highest bit.

countr_one()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > std::countr_one (
    _Tp __x ) [constexpr], [noexcept]
```

The number of contiguous one bits, starting from the lowest bit.

countr_zero()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > std::countr_zero (
    _Tp __x ) [constexpr], [noexcept]
```

The number of contiguous zero bits, starting from the lowest bit.

has_single_bit()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, bool > std::has_single_bit (
    _Tp __x ) [constexpr], [noexcept]
```

True if *x* is a power of two, false otherwise.

Referenced by [std::assume_aligned\(\)](#).

popcount()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > std::popcount (
    _Tp __x ) [constexpr], [noexcept]
```

The number of bits set in *x*.

rotl()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp > std::rotl (
    _Tp __x,
    int __s ) [constexpr], [noexcept]
```

Rotate *x* to the left by *s* bits.

rotr()

```
template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp > std::rotr (
    _Tp __x,
    int __s ) [constexpr], [noexcept]
```

Rotate *x* to the right by *s* bits.

3.11.3 Complex Numbers

Collaboration diagram for Complex Numbers:

**Classes**

- class `std::complex< _Tp >`
- class `std::complex< double >`
- class `std::complex< float >`
- class `std::complex< long double >`

Functions

- constexpr `std::complex< float >::complex` (const `complex< double >` &)
- constexpr `std::complex< float >::complex` (const `complex< long double >` &)
- constexpr `std::complex< double >::complex` (const `complex< long double >` &)
- template<typename *_Tp* >
`_Tp std::__complex_abs` (const `complex< _Tp >` &__z)
- template<typename *_Tp* >
`_Tp std::__complex_arg` (const `complex< _Tp >` &__z)
- template<typename *_Tp* >
`complex< _Tp > std::__complex_cos` (const `complex< _Tp >` &__z)
- template<typename *_Tp* >
`complex< _Tp > std::__complex_cosh` (const `complex< _Tp >` &__z)
- template<typename *_Tp* >
`complex< _Tp > std::__complex_exp` (const `complex< _Tp >` &__z)
- template<typename *_Tp* >
`complex< _Tp > std::__complex_log` (const `complex< _Tp >` &__z)

- `template<typename _Tp >`
`complex< _Tp > std::__complex_pow (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_pow_unsigned (complex< _Tp > __x, unsigned __n)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sin (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sinh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sqrt (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_tan (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_tanh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp std::abs (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::arg (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::tr1::conj (_Tp __x)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::conj (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::conj (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::cos (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::cosh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::exp (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr _Tp std::imag (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::log (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::log10 (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp constexpr std::norm (const complex< _Tp > &)`
- `constexpr complex< _Tp > & std::complex< _Tp >::operator*= (const _Tp &)`
- `template<typename _Up >`
`constexpr complex< _Tp > & std::complex< _Tp >::operator*= (const complex< _Up > &)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x)`
- `template<typename _Up >`
`constexpr complex< _Tp > & std::complex< _Tp >::operator+= (const complex< _Up > &)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x)`
- `template<typename _Up >`
`constexpr complex< _Tp > & std::complex< _Tp >::operator-= (const complex< _Up > &)`
- `constexpr complex< _Tp > & std::complex< _Tp >::operator/= (const _Tp &)`

- `template<typename _Up >`
`constexpr complex< _Tp > & std::complex< _Tp >::operator/= (const complex< _Up > &)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const complex< _Tp > &__x)`
- `constexpr complex< _Tp > & std::complex< _Tp >::operator= (const _Tp &)`
- `template<typename _Up >`
`constexpr complex< _Tp > & std::complex< _Tp >::operator= (const complex< _Up > &)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, complex< _Tp > &__x)`
- `template<typename _Tp >`
`complex< _Tp > std::polar (const _Tp &, const _Tp &=0)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::polar (const _Tp &__rho, const _Up &__theta)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const _Tp &, const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const _Tp &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const _Tp &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, int)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex< _Tp > &__x, const _Up &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex< _Tp > &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`constexpr _Tp std::real (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::sin (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sinh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sqrt (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tan (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tanh (const complex< _Tp > &)`

- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const _Tp &__y)`

3.11.3.1 Detailed Description

Classes and functions for complex numbers.

3.11.3.2 Function Documentation

abs()

```
template<typename _Tp >
_Tp std::abs (
    const complex< _Tp > & __z ) [inline]
```

Return magnitude of *z*.

Referenced by `std::chrono::abs()`, `std::binomial_distribution< _IntType >::operator()()`, and `std::poisson_distribution< _IntType >::operator()`.

arg()

```
template<typename _Tp >
_Tp std::arg (
    const complex< _Tp > & __z ) [inline]
```

Return phase angle of *z*.

conj()

```
template<typename _Tp >
constexpr complex< _Tp > std::conj (
    const complex< _Tp > & __z ) [inline], [constexpr]
```

Return complex conjugate of z.

cos()

```
template<typename _Tp >
complex< _Tp > std::cos (
    const complex< _Tp > & __z ) [inline]
```

Return complex cosine of z.

cosh()

```
template<typename _Tp >
complex< _Tp > std::cosh (
    const complex< _Tp > & __z ) [inline]
```

Return complex hyperbolic cosine of z.

exp()

```
template<typename _Tp >
complex< _Tp > std::exp (
    const complex< _Tp > & __z ) [inline]
```

Return complex base e exponential of z.

fabs()

```
template<typename _Tp >
std::complex< _Tp > std::tr1::fabs (
    const std::complex< _Tp > & __z ) [inline]
```

fabs(__z) [8.1.8].

log()

```
template<typename _Tp >
complex< _Tp > std::log (
    const complex< _Tp > & __z ) [inline]
```

Return complex natural logarithm of z.

Referenced by [std::generate_canonical\(\)](#), [std::normal_distribution<_RealType>::operator\(\)\(\)](#), [std::gamma_distribution<_RealType>::operator\(\)\(\)](#), [std::binomial_distribution<_IntType>::operator\(\)\(\)](#), [std::poisson_distribution<_IntType>::operator\(\)\(\)](#), [std::operator<<\(\)](#), [std::operator<<\(\)](#), [std::operator<<\(\)](#), and [std::operator<<\(\)](#).

log10()

```
template<typename _Tp >
complex< _Tp > std::log10 (
    const complex< _Tp > & __z ) [inline]
```

Return complex base 10 logarithm of z.

norm()

```
template<typename _Tp >
_Tp constexpr std::norm (
    const complex< _Tp > & __z ) [inline], [constexpr]
```

Return z magnitude squared.

operator*() [1/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator* (
    const _Tp & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x times y.

operator*() [2/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator* (
    const complex< _Tp > & __x,
    const _Tp & __y ) [inline], [constexpr]
```

Return new complex value x times y.

operator*() [3/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator* (
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x times y.

operator*=() [1/2]

```
template<typename _Tp >
constexpr complex< _Tp > & std::complex< _Tp >::operator*= (
    const _Tp & __t ) [constexpr]
```

Multiply this complex number by a scalar.

operator*=() [2/2]

```
template<typename _Tp >
template<typename _Up >
constexpr complex< _Tp > & std::complex< _Tp >::operator*= (
    const complex< _Up > & __z ) [constexpr]
```

Multiply this complex number by another.

operator+() [1/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator+ (
    const _Tp & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x plus y.

operator+() [2/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator+ (
    const complex< _Tp > & __x ) [inline], [constexpr]
```

Return x.

operator+() [3/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator+ (
    const complex< _Tp > & __x,
    const _Tp & __y ) [inline], [constexpr]
```

Return new complex value x plus y.

operator+() [4/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator+ (
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x plus y.

operator+=()

```
template<typename _Tp >
template<typename _Up >
constexpr complex< _Tp > & std::complex< _Tp >::operator+= (
    const complex< _Up > & __z ) [constexpr]
```

Add another complex number to this one.

operator-() [1/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator- (
    const _Tp & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x minus y.

operator-() [2/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator- (
    const complex< _Tp > & __x ) [inline], [constexpr]
```

Return complex negation of x.

operator-() [3/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator- (
    const complex< _Tp > & __x,
    const _Tp & __y ) [inline], [constexpr]
```

Return new complex value x minus y.

operator-() [4/4]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator- (
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x minus y .

operator-=()

```
template<typename _Tp >
template<typename _Up >
constexpr complex< _Tp > & std::complex< _Tp >::operator-= (
    const complex< _Up > & __z ) [constexpr]
```

Subtract another complex number from this one.

operator/() [1/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator/ (
    const _Tp & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x divided by y .

operator/() [2/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator/ (
    const complex< _Tp > & __x,
    const _Tp & __y ) [inline], [constexpr]
```

Return new complex value x divided by y .

operator/() [3/3]

```
template<typename _Tp >
constexpr complex< _Tp > std::operator/ (
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return new complex value x divided by y .

operator/=() [1/2]

```
template<typename _Tp >
constexpr complex< _Tp > & std::complex< _Tp >::operator/= (
    const _Tp & __t ) [constexpr]
```

Divide this complex number by a scalar.

operator/=() [2/2]

```
template<typename _Tp >
template<typename _Up >
constexpr complex< _Tp > & std::complex< _Tp >::operator/= (
    const complex< _Up > & __z ) [constexpr]
```

Divide this complex number by another.

operator<<()

```
template<typename _Tp , typename _CharT , class _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __os,
    const complex< _Tp > & __x )
```

Insertion operator for complex values.

operator=() [1/2]

```
template<typename _Tp >
constexpr complex< _Tp > & std::complex< _Tp >::operator= (
    const _Tp & __t ) [constexpr]
```

Assign a scalar to this complex number.

operator=() [2/2]

```
template<typename _Tp >
template<typename _Up >
constexpr complex< _Tp > & std::complex< _Tp >::operator= (
    const complex< _Up > & __z ) [constexpr]
```

Assign another complex number to this one.

operator==() [1/2]

```
template<typename _Tp >
constexpr bool std::operator==(
    const complex< _Tp > & __x,
    const _Tp & __y ) [inline], [constexpr]
```

Return true if *x* is equal to *y*.

operator==() [2/2]

```
template<typename _Tp >
constexpr bool std::operator==(
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline], [constexpr]
```

Return true if *x* is equal to *y*.

operator>>()

```
template<typename _Tp , typename _CharT , class _Traits >
basic_istream< _CharT, _Traits > & std::operator>> (
    basic_istream< _CharT, _Traits > & __is,
    complex< _Tp > & __x )
```

Extraction operator for complex values.

polar()

```
template<typename _Tp >
complex< _Tp > std::polar (
    const _Tp & __rho,
    const _Tp & __theta = 0 ) [inline]
```

Return complex with magnitude *rho* and angle *theta*.

pow() [1/5]

```
template<typename _Tp >
complex< _Tp > std::pow (
    const _Tp & __x,
    const complex< _Tp > & __y ) [inline]
```

Return x to the y 'th power.

pow() [2/5]

```
template<typename _Tp >
complex< _Tp > std::pow (
    const complex< _Tp > & __x,
    const _Tp & __y )
```

Return x to the y 'th power.

pow() [3/5]

```
template<typename _Tp >
complex< _Tp > std::pow (
    const complex< _Tp > & __x,
    const complex< _Tp > & __y ) [inline]
```

Return x to the y 'th power.

pow() [4/5]

```
template<typename _Tp >
complex< _Tp > std::pow (
    const complex< _Tp > & __x,
    int __n ) [inline]
```

Return x to the y 'th power.

Referenced by `std::gamma_distribution< _RealType >::operator()()`, and `std::operator<<()`.

pow() [5/5]

```
template<typename _Tp , typename _Up >
std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (
    const std::complex< _Tp > & __x,
    const _Up & __y ) [inline]
```

Additional overloads [8.1.9].

sin()

```
template<typename _Tp >
complex< _Tp > std::sin (
    const complex< _Tp > & __z ) [inline]
```

Return complex sine of z .

sinh()

```
template<typename _Tp >
complex< _Tp > std::sinh (
    const complex< _Tp > & __z ) [inline]
```

Return complex hyperbolic sine of z .

sqrt()

```
template<typename _Tp >
complex< _Tp > std::sqrt (
    const complex< _Tp > & __z ) [inline]
```

Return complex square root of z.

Referenced by [std::student_t_distribution< _RealType >::operator\(\)\(\)](#), and [std::normal_distribution< _RealType >::operator\(\)\(\)](#).

tan()

```
template<typename _Tp >
complex< _Tp > std::tan (
    const complex< _Tp > & __z ) [inline]
```

Return complex tangent of z.

Referenced by [std::operator<<\(\)](#).

tanh()

```
template<typename _Tp >
complex< _Tp > std::tanh (
    const complex< _Tp > & __z ) [inline]
```

Return complex hyperbolic tangent of z.

3.11.4 Decimal Floating-Point Arithmetic

Collaboration diagram for Decimal Floating-Point Arithmetic:

**Namespaces**

- namespace [std::decimal](#)

Classes

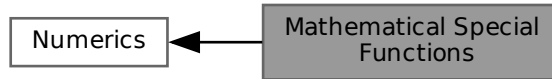
- class [std::decimal::decimal128](#)
- class [std::decimal::decimal32](#)
- class [std::decimal::decimal64](#)

3.11.4.1 Detailed Description

Classes and functions for decimal floating-point arithmetic.

3.11.5 Mathematical Special Functions

Collaboration diagram for Mathematical Special Functions:



Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_ai (_Tp __x)`
- `float __gnu_cxx::airy_aif (float __x)`
- `long double __gnu_cxx::airy_ail (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_bi (_Tp __x)`
- `float __gnu_cxx::airy_bif (float __x)`
- `long double __gnu_cxx::airy_bil (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float std::assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double std::assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float std::assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double std::assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type std::beta (_Tpa __a, _Tpb __b)`
- `float std::betaf (float __a, float __b)`
- `long double std::betal (long double __a, long double __b)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_1 (_Tp __k)`
- `float std::comp_ellint_1f (float __k)`
- `long double std::comp_ellint_1l (long double __k)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_2 (_Tp __k)`
- `float std::comp_ellint_2f (float __k)`
- `long double std::comp_ellint_2l (long double __k)`
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::comp_ellint_3 (_Tp __k, _Tpn __nu)`
- `float std::comp_ellint_3f (float __k, float __nu)`
- `long double std::comp_ellint_3l (long double __k, long double __nu)`
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type __gnu_cxx::conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x)`
- `float __gnu_cxx::conf_hypergf (float __a, float __c, float __x)`
- `long double __gnu_cxx::conf_hypergl (long double __a, long double __c, long double __x)`

- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_i (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_if (float __nu, float __x)`
- `long double std::cyl_bessel_il (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_j (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_jf (float __nu, float __x)`
- `long double std::cyl_bessel_jl (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_k (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_kf (float __nu, float __x)`
- `long double std::cyl_bessel_kl (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_neumann (_Tpnu __nu, _Tp __x)`
- `float std::cyl_neumannf (float __nu, float __x)`
- `long double std::cyl_neumannl (long double __nu, long double __x)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_1 (_Tp __k, _Tpp __phi)`
- `float std::ellint_1f (float __k, float __phi)`
- `long double std::ellint_1l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_2 (_Tp __k, _Tpp __phi)`
- `float std::ellint_2f (float __k, float __phi)`
- `long double std::ellint_2l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpn, typename _Tpp >`
`__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type std::ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi)`
- `float std::ellint_3f (float __k, float __nu, float __phi)`
- `long double std::ellint_3l (long double __k, long double __nu, long double __phi)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::expint (_Tp __x)`
- `float std::expintf (float __x)`
- `long double std::expintl (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::hermite (unsigned int __n, _Tp __x)`
- `float std::hermitef (unsigned int __n, float __x)`
- `long double std::hermitel (unsigned int __n, long double __x)`
- `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type __gnu_cxx::hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)`
- `float __gnu_cxx::hypergf (float __a, float __b, float __c, float __x)`
- `long double __gnu_cxx::hypergl (long double __a, long double __b, long double __c, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::laguerre (unsigned int __n, _Tp __x)`
- `float std::laguerref (unsigned int __n, float __x)`
- `long double std::laguerrel (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::legendre (unsigned int __l, _Tp __x)`
- `float std::legendref (unsigned int __l, float __x)`
- `long double std::legendrel (unsigned int __l, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::riemann_zeta (_Tp __s)`
- `float std::riemann_zetaf (float __s)`

- long double [std::riemann_zeta](#) (long double __s)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [std::sph_bessel](#) (unsigned int __n, _Tp __x)
- float [std::sph_besself](#) (unsigned int __n, float __x)
- long double [std::sph_bessell](#) (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [std::sph_legendre](#) (unsigned int __l, unsigned int __m, _Tp __theta)
- float [std::sph_legendref](#) (unsigned int __l, unsigned int __m, float __theta)
- long double [std::sph_legendrel](#) (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [std::sph_neumann](#) (unsigned int __n, _Tp __x)
- float [std::sph_neumannf](#) (unsigned int __n, float __x)
- long double [std::sph_neumannl](#) (unsigned int __n, long double __x)

3.11.5.1 Detailed Description

3.11.5.2 Mathematical Special Functions

A collection of advanced mathematical special functions, defined by ISO/IEC IS 29124 and then added to ISO C++ 2017.

Introduction and History

The first significant library upgrade on the road to C++2011, [TR1](#), included a set of 23 mathematical functions that significantly extended the standard transcendental functions inherited from C and declared in `<cmath>`.

Although most components from TR1 were eventually adopted for C++11 these math functions were left behind out of concern for implementability. The math functions were published as a separate international standard [IS 29124 - Extensions to the C++ Library to Support Mathematical Special Functions](#).

For C++17 these functions were incorporated into the main standard.

Contents

The following functions are implemented in namespace `std`:

- `assoc_laguerre` - Associated Laguerre functions
- `assoc_legendre` - Associated Legendre functions
- `beta` - Beta functions
- `comp_ellint_1` - Complete elliptic functions of the first kind
- `comp_ellint_2` - Complete elliptic functions of the second kind
- `comp_ellint_3` - Complete elliptic functions of the third kind
- `cyl_bessel_i` - Regular modified cylindrical Bessel functions
- `cyl_bessel_j` - Cylindrical Bessel functions of the first kind
- `cyl_bessel_k` - Irregular modified cylindrical Bessel functions
- `cyl_neumann` - Cylindrical Neumann functions or Cylindrical Bessel functions of the second kind
- `ellint_1` - Incomplete elliptic functions of the first kind
- `ellint_2` - Incomplete elliptic functions of the second kind
- `ellint_3` - Incomplete elliptic functions of the third kind

- `expint` - The exponential integral
- `hermite` - Hermite polynomials
- `laguerre` - Laguerre functions
- `legendre` - Legendre polynomials
- `riemann_zeta` - The Riemann zeta function
- `sph_bessel` - Spherical Bessel functions
- `sph_legendre` - Spherical Legendre functions
- `sph_neumann` - Spherical Neumann functions

The hypergeometric functions were stricken from the TR29124 and C++17 versions of this math library because of implementation concerns. However, since they were in the TR1 version and since they are popular we kept them as an extension in namespace `__gnu_cxx`:

- [conf_hyperg](#) - Confluent hypergeometric functions
- [hyperg](#) - Hypergeometric functions

Argument Promotion

The arguments supplied to the non-suffixed functions will be promoted according to the following rules:

1. If any argument intended to be floating point is given an integral value That integral value is promoted to double.
2. All floating point arguments are promoted up to the largest floating point precision among them.

NaN Arguments

If any of the floating point arguments supplied to these functions is invalid or NaN (`std::numeric_limits<Tp>::quiet_NaN`), the value NaN is returned.

Implementation

We strive to implement the underlying math with type generic algorithms to the greatest extent possible. In practice, the functions are thin wrappers that dispatch to function templates. Type dependence is controlled with `std::numeric_limits` and functions thereof.

We don't promote `float` to `double` or `double` to `long double` reflexively. The goal is for `float` functions to operate more quickly, at the cost of `float` accuracy and possibly a smaller domain of validity. Similarly, `long double` should give you more dynamic range and slightly more precision than `double` on many systems.

Testing

These functions have been tested against equivalent implementations from the [Gnu Scientific Library](#), [GSL](#) and [Boost](#) and the ratio

$$\frac{|f - f_{test}|}{|f_{test}|}$$

is generally found to be within 10^{-15} for 64-bit double on linux-x86_64 systems over most of the ranges of validity.

Todo Provide accuracy comparisons on a per-function basis for a small number of targets.

General Bibliography

See also

Abramowitz and Stegun: Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables
 Edited by Milton Abramowitz and Irene A. Stegun, National Bureau of Standards Applied Mathematics Series - 55
 Issued June 1964, Tenth Printing, December 1972, with corrections Electronic versions of A&S abound including
 both pdf and navigable html.

for example <http://people.math.sfu.ca/~cbm/aands/>

The old A&S has been redone as the NIST Digital Library of Mathematical Functions: <http://dlmf.nist.gov/> This version is far more navigable and includes more recent work.

An Atlas of Functions: with Equator, the Atlas Function Calculator 2nd Edition, by Oldham, Keith B., Myland, Jan, Spanier, Jerome

Asymptotics and Special Functions by Frank W. J. Olver, Academic Press, 1974

Numerical Recipes in C, The Art of Scientific Computing, by William H. Press, Second Ed., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, Cambridge University Press, 1992

The Special Functions and Their Approximations: Volumes 1 and 2, by Yudell L. Luke, Academic Press, 1969

3.11.5.3 Function Documentation

airy_ai()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_ai (
    _Tp __x ) [inline]
```

Return the Airy function $Ai(x)$ of real argument x.

airy_aif()

```
float __gnu_cxx::airy_aif (
    float __x ) [inline]
```

Return the Airy function $Ai(x)$ of float argument x.

airy_ail()

```
long double __gnu_cxx::airy_ail (
    long double __x ) [inline]
```

Return the Airy function $Ai(x)$ of long double argument x.

airy_bi()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_bi (
    _Tp __x ) [inline]
```

Return the Airy function $Bi(x)$ of real argument x.

airy_bif()

```
float __gnu_cxx::airy_bif (
    float __x ) [inline]
```

Return the Airy function $Bi(x)$ of float argument x.

airy_bil()

```
long double __gnu_cxx::airy_bil (
    long double __x ) [inline]
```

Return the Airy function $Bi(x)$ of long double argument x .

assoc_laguerre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::assoc_laguerre (
    unsigned int __n,
    unsigned int __m,
    _Tp __x ) [inline]
```

Return the associated Laguerre polynomial of nonnegative order n , nonnegative degree m and real argument $x \leftarrow : L_n^m(x)$.

The associated Laguerre function of real degree α , $L_n^\alpha(x)$, is defined by

$$L_n^\alpha(x) = \frac{(\alpha+1)_n}{n!} {}_1F_1(-n; \alpha+1; x)$$

where $(\alpha)_n$ is the Pochhammer symbol and ${}_1F_1(a; c; x)$ is the confluent hypergeometric function.

The associated Laguerre polynomial is defined for integral degree $\alpha = m$ by:

$$L_n^m(x) = (-1)^m \frac{d^m}{dx^m} L_{n+m}(x)$$

where the Laguerre polynomial is defined by:

$$L_n(x) = \frac{e^x}{n!} \frac{d^n}{dx^n} (x^n e^{-x})$$

and $x \geq 0$.

See also

laguerre for details of the Laguerre function of degree n

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The order of the Laguerre function, <code>__n</code> ≥ 0 .
<code>__m</code>	The degree of the Laguerre function, <code>__m</code> ≥ 0 .
<code>__x</code>	The argument of the Laguerre function, <code>__x</code> ≥ 0 .

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

assoc_laguerref()

```
float std::assoc_laguerref (
    unsigned int __n,
    unsigned int __m,
    float __x ) [inline]
```

Return the associated Laguerre polynomial of order n , degree m : $L_n^m(x)$ for `float` argument.

See also

`assoc_laguerre` for more details.

assoc_laguerrel()

```
long double std::assoc_laguerrel (
    unsigned int __n,
    unsigned int __m,
    long double __x ) [inline]
```

Return the associated Laguerre polynomial of order n , degree m : $L_n^m(x)$.

See also

`assoc_laguerre` for more details.

assoc_legendre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::assoc_legendre (
    unsigned int __l,
    unsigned int __m,
    _Tp __x ) [inline]
```

Return the associated Legendre function of degree l and order m .

The associated Legendre function is derived from the Legendre function $P_l(x)$ by the Rodrigues formula:

$$P_l^m(x) = (1 - x^2)^{m/2} \frac{d^m}{dx^m} P_l(x)$$

See also

`legendre` for details of the Legendre function of degree l

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__l</code>	The degree <code>__l >= 0</code> .
<code>__m</code>	The order <code>__m <= l</code> .
<code>__x</code>	The argument, <code>abs (__x) <= 1</code> .

Exceptions

<code>std::domain_error</code>	if <code>abs(__x) > 1.</code>
--------------------------------	----------------------------------

assoc_legendref()

```
float std::assoc_legendref (
    unsigned int __l,
    unsigned int __m,
    float __x ) [inline]
```

Return the associated Legendre function of degree `l` and order `m` for `float` argument.

See also

`assoc_legendre` for more details.

assoc_legendrel()

```
long double std::assoc_legendrel (
    unsigned int __l,
    unsigned int __m,
    long double __x ) [inline]
```

Return the associated Legendre function of degree `l` and order `m`.

See also

`assoc_legendre` for more details.

beta()

```
template<typename _Tpa , typename _Tpb >
__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type std::beta (
    _Tpa __a,
    _Tpb __b ) [inline]
```

Return the beta function, $B(a, b)$, for real parameters `a`, `b`.

The beta function is defined by

$$B(a, b) = \int_0^1 t^{a-1} (1-t)^{b-1} dt = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

where $a > 0$ and $b > 0$

Template Parameters

<code>_Tpa</code>	The floating-point type of the parameter <code>__a</code> .
<code>_Tpb</code>	The floating-point type of the parameter <code>__b</code> .

Parameters

<code>__a</code>	The first argument of the beta function, <code>__a > 0</code> .
<code>__b</code>	The second argument of the beta function, <code>__b > 0</code> .

Exceptions

<code>std::domain_error</code>	if <code>__a < 0</code> or <code>__b < 0</code> .
--------------------------------	---

betaf()

```
float std::betaf (
    float __a,
    float __b ) [inline]
```

Return the beta function, $B(a, b)$, for `float` parameters `a`, `b`.

See also

`beta` for more details.

betal()

```
long double std::betal (
    long double __a,
    long double __b ) [inline]
```

Return the beta function, $B(a, b)$, for long double parameters `a`, `b`.

See also

`beta` for more details.

comp_ellint_1()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_1 (
    _Tp __k ) [inline]
```

Return the complete elliptic integral of the first kind $K(k)$ for real modulus `k`.

The complete elliptic integral of the first kind is defined as

$$K(k) = F(k, \pi/2) = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1 - k^2 \sin^2 \theta}}$$

where $F(k, \phi)$ is the incomplete elliptic integral of the first kind and the modulus $|k| \leq 1$.

See also

`ellint_1` for details of the incomplete elliptic function of the first kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
------------------	---

Parameters

<code>__k</code>	The modulus, <code>abs(__k) <= 1</code>
------------------	--

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

comp_ellint_1f()

```
float std::comp_ellint_1f (
    float __k ) [inline]
```

Return the complete elliptic integral of the first kind $E(k)$ for `float` modulus `k`.

See also

`comp_ellint_1` for details.

comp_ellint_1l()

```
long double std::comp_ellint_1l (
    long double __k ) [inline]
```

Return the complete elliptic integral of the first kind $E(k)$ for `long double` modulus `k`.

See also

`comp_ellint_1` for details.

comp_ellint_2()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_2 (
    _Tp __k ) [inline]
```

Return the complete elliptic integral of the second kind $E(k)$ for real modulus `k`.
The complete elliptic integral of the second kind is defined as

$$E(k) = E(k, \pi/2) = \int_0^{\pi/2} \sqrt{1 - k^2 \sin^2 \theta}$$

where $E(k, \phi)$ is the incomplete elliptic integral of the second kind and the modulus $|k| \leq 1$.

See also

`ellint_2` for details of the incomplete elliptic function of the second kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
------------------	---

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
------------------	---

Exceptions

<code>std::domain_error</code>	if <code>abs(__k) > 1</code> .
--------------------------------	-----------------------------------

comp_ellint_2f()

```
float std::comp_ellint_2f (
    float __k ) [inline]
```

Return the complete elliptic integral of the second kind $E(k)$ for `float` modulus `k`.

See also

`comp_ellint_2` for details.

comp_ellint_2l()

```
long double std::comp_ellint_2l (
    long double __k ) [inline]
```

Return the complete elliptic integral of the second kind $E(k)$ for long double modulus `k`.

See also

`comp_ellint_2` for details.

comp_ellint_3()

```
template<typename _Tp , typename _Tpn >
__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::comp_ellint_3 (
    _Tp __k,
    _Tpn __nu ) [inline]
```

Return the complete elliptic integral of the third kind $\Pi(k, \nu) = \Pi(k, \nu, \pi/2)$ for real modulus `k`.

The complete elliptic integral of the third kind is defined as

$$\Pi(k, \nu) = \Pi(k, \nu, \pi/2) = \int_0^{\pi/2} \frac{d\theta}{(1 - \nu \sin^2 \theta) \sqrt{1 - k^2 \sin^2 \theta}}$$

where $\Pi(k, \nu, \phi)$ is the incomplete elliptic integral of the second kind and the modulus $|k| \leq 1$.

See also

`ellint_3` for details of the incomplete elliptic function of the third kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpn</code>	The floating-point type of the argument <code>__nu</code> .

Parameters

<code>__k</code>	The modulus, <code>abs(__k) <= 1</code>
<code>__nu</code>	The argument

Exceptions

<code>std::domain_error</code>	if <code>abs(__k) > 1</code> .
--------------------------------	-----------------------------------

comp_ellint_3f()

```
float std::comp_ellint_3f (
    float __k,
    float __nu ) [inline]
```

Return the complete elliptic integral of the third kind $\Pi(k, \nu)$ for `float` modulus `k`.

See also

`comp_ellint_3` for details.

comp_ellint_3l()

```
long double std::comp_ellint_3l (
    long double __k,
    long double __nu ) [inline]
```

Return the complete elliptic integral of the third kind $\Pi(k, \nu)$ for `long double` modulus `k`.

See also

`comp_ellint_3` for details.

conf_hyperg()

```
template<typename _Tpa , typename _Tpc , typename _Tp >
__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type __gnu_cxx::conf_hyperg (
    _Tpa __a,
    _Tpc __c,
    _Tp __x ) [inline]
```

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of real numeratorial parameter `a`, denominatorial parameter `c`, and argument `x`.

The confluent hypergeometric function is defined by

$${}_1F_1(a; c; x) = \sum_{n=0}^{\infty} \frac{(a)_n x^n}{(c)_n n!}$$

where the Pochhammer symbol is $(x)_k = (x)(x+1)\dots(x+k-1)$, $(x)_0 = 1$

Parameters

<code>↔ _a</code>	The numeratorial parameter
<code>↔ _c</code>	The denominatorial parameter
<code>↔ _x</code>	The argument

conf_hypergf()

```
float __gnu_cxx::conf_hypergf (
    float __a,
    float __c,
    float __x ) [inline]
```

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of `float` numeratorial parameter `a`, denominatorial parameter `c`, and argument `x`.

See also

`conf_hyperg` for details.

conf_hypergl()

```
long double __gnu_cxx::conf_hypergl (
    long double __a,
    long double __c,
    long double __x ) [inline]
```

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of `long double` numeratorial parameter `a`, denominatorial parameter `c`, and argument `x`.

See also

`conf_hyperg` for details.

cyl_bessel_i()

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_i (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

Return the regular modified Bessel function $I_\nu(x)$ for real order ν and argument $x \geq 0$. The regular modified cylindrical Bessel function is:

$$I_\nu(x) = i^{-\nu} J_\nu(ix) = \sum_{k=0}^{\infty} \frac{(x/2)^{\nu+2k}}{k! \Gamma(\nu + k + 1)}$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

cyl_bessel_if()

```
float std::cyl_bessel_if (
    float __nu,
    float __x ) [inline]
```

Return the regular modified Bessel function $I_\nu(x)$ for float order ν and argument $x \geq 0$.

See also

`cyl_bessel_i` for setails.

cyl_bessel_il()

```
long double std::cyl_bessel_il (
    long double __nu,
    long double __x ) [inline]
```

Return the regular modified Bessel function $I_\nu(x)$ for long double order ν and argument $x \geq 0$.

See also

`cyl_bessel_i` for setails.

cyl_bessel_j()

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_j (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

Return the Bessel function $J_\nu(x)$ of real order ν and argument $x \geq 0$.

The cylindrical Bessel function is:

$$J_\nu(x) = \sum_{k=0}^{\infty} \frac{(-1)^k (x/2)^{\nu+2k}}{k! \Gamma(\nu + k + 1)}$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

cyl_bessel_jf()

```
float std::cyl_bessel_jf (
```

```
float __nu,
float __x ) [inline]
```

Return the Bessel function of the first kind $J_\nu(x)$ for `float` order ν and argument $x \geq 0$.

See also

`cyl_bessel_j` for details.

`cyl_bessel_jl()`

```
long double std::cyl_bessel_jl (
    long double __nu,
    long double __x ) [inline]
```

Return the Bessel function of the first kind $J_\nu(x)$ for `long double` order ν and argument $x \geq 0$.

See also

`cyl_bessel_j` for details.

`cyl_bessel_k()`

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_k (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

Return the irregular modified Bessel function $K_\nu(x)$ of real order ν and argument x .
The irregular modified Bessel function is defined by:

$$K_\nu(x) = \frac{\pi}{2} \frac{I_{-\nu}(x) - I_\nu(x)}{\sin \nu\pi}$$

where for integral $\nu = n$ a limit is taken: $\lim_{\nu \rightarrow n}$. For negative argument we have simply:

$$K_{-\nu}(x) = K_\nu(x)$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

cyl_bessel_kf()

```
float std::cyl_bessel_kf (
    float __nu,
    float __x ) [inline]
```

Return the irregular modified Bessel function $K_\nu(x)$ for float order ν and argument $x \geq 0$.

See also

`cyl_bessel_k` for setails.

cyl_bessel_kl()

```
long double std::cyl_bessel_kl (
    long double __nu,
    long double __x ) [inline]
```

Return the irregular modified Bessel function $K_\nu(x)$ for long double order ν and argument $x \geq 0$.

See also

`cyl_bessel_k` for setails.

cyl_neumann()

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_neumann (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

Return the Neumann function $N_\nu(x)$ of real order ν and argument $x \geq 0$.

The Neumann function is defined by:

$$N_\nu(x) = \frac{J_\nu(x) \cos \nu\pi - J_{-\nu}(x)}{\sin \nu\pi}$$

where $x \geq 0$ and for integral order $\nu = n$ a limit is taken: $\lim_{\nu \rightarrow n}$.

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

cyl_neumannf()

```
float std::cyl_neumannf (
```

```
float __nu,
float __x ) [inline]
```

Return the Neumann function $N_\nu(x)$ of `float` order ν and argument x .

See also

`cyl_neumann` for details.

`cyl_neumannl()`

```
long double std::cyl_neumannl (
    long double __nu,
    long double __x ) [inline]
```

Return the Neumann function $N_\nu(x)$ of `long double` order ν and argument x .

See also

`cyl_neumann` for details.

`ellint_1()`

```
template<typename _Tp , typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_1 (
    _Tp __k,
    _Tpp __phi ) [inline]
```

Return the incomplete elliptic integral of the first kind $F(k, \phi)$ for `real` modulus k and angle ϕ .
The incomplete elliptic integral of the first kind is defined as

$$F(k, \phi) = \int_0^\phi \frac{d\theta}{\sqrt{1 - k^2 \sin^2 \theta}}$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the first kind, $K(k)$.

See also

`comp_ellint_1`.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__phi</code>	The integral limit argument in radians

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

ellint_1f()

```
float std::ellint_1f (
    float __k,
    float __phi ) [inline]
```

Return the incomplete elliptic integral of the first kind $E(k, \phi)$ for `float` modulus k and angle ϕ .

See also

`ellint_1` for details.

ellint_1l()

```
long double std::ellint_1l (
    long double __k,
    long double __phi ) [inline]
```

Return the incomplete elliptic integral of the first kind $E(k, \phi)$ for `long double` modulus k and angle ϕ .

See also

`ellint_1` for details.

ellint_2()

```
template<typename _Tp , typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_2 (
    _Tp __k,
    _Tpp __phi ) [inline]
```

Return the incomplete elliptic integral of the second kind $E(k, \phi)$.

The incomplete elliptic integral of the second kind is defined as

$$E(k, \phi) = \int_0^\phi \sqrt{1 - k^2 \sin^2 \theta}$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the second kind, $E(k)$.

See also

`comp_ellint_2`.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__phi</code>	The integral limit argument in radians

Returns

The elliptic function of the second kind.

Exceptions

<code>std::domain_error</code>	if <code>abs(__k) > 1</code> .
--------------------------------	-----------------------------------

ellint_2f()

```
float std::ellint_2f (
    float __k,
    float __phi ) [inline]
```

Return the incomplete elliptic integral of the second kind $E(k, \phi)$ for `float` argument.

See also

`ellint_2` for details.

ellint_2l()

```
long double std::ellint_2l (
    long double __k,
    long double __phi ) [inline]
```

Return the incomplete elliptic integral of the second kind $E(k, \phi)$.

See also

`ellint_2` for details.

ellint_3()

```
template<typename _Tp , typename _Tpn , typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type std::ellint_3 (
    _Tp __k,
    _Tpn __nu,
    _Tpp __phi ) [inline]
```

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$.

The incomplete elliptic integral of the third kind is defined by:

$$\Pi(k, \nu, \phi) = \int_0^\phi \frac{d\theta}{(1 - \nu \sin^2 \theta) \sqrt{1 - k^2 \sin^2 \theta}}$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the third kind, $\Pi(k, \nu)$.

See also

`comp_ellint_3`.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpn</code>	The floating-point type of the argument <code>__nu</code> .
<code>_Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__nu</code>	The second argument
<code>__phi</code>	The integral limit argument in radians

Returns

The elliptic function of the third kind.

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

ellint_3f()

```
float std::ellint_3f (
    float __k,
    float __nu,
    float __phi ) [inline]
```

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$ for `float` argument.

See also

`ellint_3` for details.

ellint_3l()

```
long double std::ellint_3l (
    long double __k,
    long double __nu,
    long double __phi ) [inline]
```

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$.

See also

`ellint_3` for details.

expint()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::expint (
    _Tp __x ) [inline]
```

Return the exponential integral $Ei(x)$ for `real` argument `x`.

The exponential integral is given by

$$Ei(x) = - \int_{-x}^{\infty} \frac{e^t}{t} dt$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__x</code>	The argument of the exponential integral function.
------------------	--

expintf()

```
float std::expintf (
    float __x ) [inline]
```

Return the exponential integral $Ei(x)$ for `float` argument `x`.

See also

`expint` for details.

expintl()

```
long double std::expintl (
    long double __x ) [inline]
```

Return the exponential integral $Ei(x)$ for `long double` argument `x`.

See also

`expint` for details.

hermite()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::hermite (
    unsigned int __n,
    _Tp __x ) [inline]
```

Return the Hermite polynomial $H_n(x)$ of order `n` and `real` argument `x`.

The Hermite polynomial is defined by:

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2}$$

The Hermite polynomial obeys a reflection formula:

$$H_n(-x) = (-1)^n H_n(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The order
<code>__x</code>	The argument

hermitef()

```
float std::hermitef (
    unsigned int __n,
    float __x ) [inline]
```

Return the Hermite polynomial $H_n(x)$ of nonnegative order n and float argument x.

See also

hermite for details.

hermitel()

```
long double std::hermitel (
    unsigned int __n,
    long double __x ) [inline]
```

Return the Hermite polynomial $H_n(x)$ of nonnegative order n and long double argument x.

See also

hermite for details.

hyperg()

```
template<typename _Tpa , typename _Tpb , typename _Tpc , typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type __gnu_cxx::hyperg (
    _Tpa __a,
    _Tpb __b,
    _Tpc __c,
    _Tp __x ) [inline]
```

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of real numeratorial parameters a and b, denominatorial parameter c, and argument x.

The hypergeometric function is defined by

$${}_2F_1(a; c; x) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n x^n}{(c)_n n!}$$

where the Pochhammer symbol is $(x)_k = (x)(x+1)\dots(x+k-1)$, $(x)_0 = 1$

Parameters

\leftrightarrow _a	The first numeratorial parameter
\leftrightarrow _b	The second numeratorial parameter
\leftrightarrow _c	The denominatorial parameter
\leftrightarrow _x	The argument

hypergf()

```
float __gnu_cxx::hypergf (
```

```
float __a,
float __b,
float __c,
float __x ) [inline]
```

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of @ float numeratorial parameters a and b, denominatorial parameter c, and argument x.

See also

hyperg for details.

hypergl()

```
long double __gnu_cxx::hypergl (
    long double __a,
    long double __b,
    long double __c,
    long double __x ) [inline]
```

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of long double numeratorial parameters a and b, denominatorial parameter c, and argument x.

See also

hyperg for details.

laguerre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::laguerre (
    unsigned int __n,
    _Tp __x ) [inline]
```

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree n and real argument $x \geq 0$.

The Laguerre polynomial is defined by:

$$L_n(x) = \frac{e^x}{n!} \frac{d^n}{dx^n} (x^n e^{-x})$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The nonnegative order
<code>__x</code>	The argument <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x</code> < 0 .
--------------------------------	-----------------------------

laguerref()

```
float std::laguerref (
    unsigned int __n,
    float __x ) [inline]
```

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree `n` and `float` argument $x \geq 0$.

See also

`laguerre` for more details.

laguerrel()

```
long double std::laguerrel (
    unsigned int __n,
    long double __x ) [inline]
```

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree `n` and `long double` argument $x \geq 0$.

See also

`laguerre` for more details.

legendre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::legendre (
    unsigned int __l,
    _Tp __x ) [inline]
```

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and real argument $|x| \leq 1$.

The Legendre function of order l and argument x , $P_l(x)$, is defined by:

$$P_l(x) = \frac{1}{2^l l!} \frac{d^l}{dx^l} (x^2 - 1)^l$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__l</code>	The degree $l \geq 0$
<code>__x</code>	The argument $\text{abs}(\text{__x}) \leq 1$

Exceptions

<code>std::domain_error</code>	if $\text{abs}(\text{__x}) > 1$
--------------------------------	---------------------------------

legendref()

```
float std::legendref (
    unsigned int __l,
    float __x ) [inline]
```

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and float argument $|x| \leq 0$.

See also

legendre for more details.

legendrel()

```
long double std::legendrel (
    unsigned int __l,
    long double __x ) [inline]
```

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and long double argument $|x| \leq 0$.

See also

legendre for more details.

riemann_zeta()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::riemann_zeta (
    _Tp __s ) [inline]
```

Return the Riemann zeta function $\zeta(s)$ for real argument s .

The Riemann zeta function is defined by:

$$\zeta(s) = \sum_{k=1}^{\infty} k^{-s} \text{ for } s > 1$$

and

$$\zeta(s) = \frac{1}{1-2^{1-s}} \sum_{k=1}^{\infty} (-1)^{k-1} k^{-s} \text{ for } 0 \leq s \leq 1$$

For $s < 1$ use the reflection formula:

$$\zeta(s) = 2^s \pi^{s-1} \sin\left(\frac{\pi s}{2}\right) \Gamma(1-s) \zeta(1-s)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__s</code> .
------------------	--

Parameters

<code>__s</code>	The argument $s \neq 1$
------------------	-------------------------

riemann_zetaf()

```
float std::riemann_zetaf (
```

```
float __s ) [inline]
```

Return the Riemann zeta function $\zeta(s)$ for float argument s .

See also

`riemann_zeta` for more details.

riemann_zetal()

```
long double std::riemann_zetal (
    long double __s ) [inline]
```

Return the Riemann zeta function $\zeta(s)$ for long double argument s .

See also

`riemann_zeta` for more details.

sph_bessel()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::sph_bessel (
    unsigned int __n,
    _Tp __x ) [inline]
```

Return the spherical Bessel function $j_n(x)$ of nonnegative order n and real argument $x \geq 0$. The spherical Bessel function is defined by:

$$j_n(x) = \left(\frac{\pi}{2x}\right)^{1/2} J_{n+1/2}(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The integral order $n \geq 0$
<code>__x</code>	The real argument $x \geq 0$

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

sph_besself()

```
float std::sph_besself (
    unsigned int __n,
    float __x ) [inline]
```

Return the spherical Bessel function $j_n(x)$ of nonnegative order n and float argument $x \geq 0$.

See also

`sph_bessel` for more details.

`sph_bessell()`

```
long double std::sph_bessell (
    unsigned int __n,
    long double __x ) [inline]
```

Return the spherical Bessel function $j_n(x)$ of nonnegative order `n` and `long double` argument $x \geq 0$.

See also

`sph_bessel` for more details.

`sph_legendre()`

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::sph_legendre (
    unsigned int __l,
    unsigned int __m,
    _Tp __theta ) [inline]
```

Return the spherical Legendre function of nonnegative integral degree `l` and order `m` and real angle θ in radians. The spherical Legendre function is defined by

$$Y_l^m(\theta, \phi) = (-1)^m \left[\frac{(2l+1)}{4\pi} \frac{(l-m)!}{(l+m)!} \right] P_l^m(\cos \theta) \exp^{im\phi}$$

Template Parameters

<code>_Tp</code>	The floating-point type of the angle <code>__theta</code> .
------------------	---

Parameters

<code>__l</code>	The order <code>__l</code> ≥ 0
<code>__m</code>	The degree <code>__m</code> ≥ 0 and <code>__m</code> \leq <code>__l</code>
<code>__theta</code>	The radian polar angle argument

`sph_legendref()`

```
float std::sph_legendref (
    unsigned int __l,
    unsigned int __m,
    float __theta ) [inline]
```

Return the spherical Legendre function of nonnegative integral degree `l` and order `m` and float angle θ in radians.

See also

`sph_legendre` for details.

sph_legendrel()

```
long double std::sph_legendrel (
    unsigned int __l,
    unsigned int __m,
    long double __theta ) [inline]
```

Return the spherical Legendre function of nonnegative integral degree l and order m and long double angle θ in radians.

See also

`sph_legendre` for details.

sph_neumann()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::sph_neumann (
    unsigned int __n,
    _Tp __x ) [inline]
```

Return the spherical Neumann function of integral order $n \geq 0$ and real argument $x \geq 0$. The spherical Neumann function is defined by

$$n_n(x) = \left(\frac{\pi}{2x}\right)^{1/2} N_{n+1/2}(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>↔ __n</code>	The integral order $n \geq 0$
<code>↔ __x</code>	The real argument <code>__x</code> ≥ 0

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

sph_neumannf()

```
float std::sph_neumannf (
    unsigned int __n,
    float __x ) [inline]
```

Return the spherical Neumann function of integral order $n \geq 0$ and float argument $x \geq 0$.

See also

`sph_neumann` for details.

sph_neumannl()

```
long double std::sph_neumannl (
    unsigned int __n,
    long double __x) [inline]
```

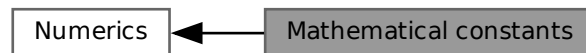
Return the spherical Neumann function of integral order $n \geq 0$ and long double $x \geq 0$.

See also

sph_neumann for details.

3.11.6 Mathematical constants

Collaboration diagram for Mathematical constants:

**Namespaces**

- namespace [std::numbers](#)

3.11.6.1 Detailed Description**3.11.7 Numeric Arrays**

Collaboration diagram for Numeric Arrays:

**Classes**

- class [std::gslice](#)
- class [std::gslice_array< _Tp >](#)
- class [std::indirect_array< _Tp >](#)
- class [std::mask_array< _Tp >](#)
- class [std::slice](#)
- class [std::slice_array< _Tp >](#)
- class [std::valarray< _Tp >](#)

Functions

- `std::gslice::gslice ()`
- `std::gslice::gslice (const gslice &)`
- `std::gslice::gslice (size_t __o, const valarray< size_t > &__l, const valarray< size_t > &__s)`
- `std::gslice_array< _Tp >::gslice_array (const gslice_array &)`
- `std::indirect_array< _Tp >::indirect_array (const indirect_array &)`
- `std::mask_array< _Tp >::mask_array (const mask_array &)`
- `std::slice::slice ()`
- `std::slice::slice (size_t __o, size_t __d, size_t __s)`
- `std::slice_array< _Tp >::slice_array (const slice_array &)`
- `std::valarray< _Tp >::valarray () noexcept`
- `template<class _Dom >`
`std::valarray< _Tp >::valarray (const Expr< _Dom, _Tp > &__e)`
- `std::valarray< _Tp >::valarray (const _Tp &, size_t)`
- `template<typename _Tp >`
`std::valarray< _Tp >::valarray (const _Tp *__restrict __p, size_t __n)`
- `std::valarray< _Tp >::valarray (const gslice_array< _Tp > &)`
- `std::valarray< _Tp >::valarray (const indirect_array< _Tp > &)`
- `std::valarray< _Tp >::valarray (const mask_array< _Tp > &)`
- `std::valarray< _Tp >::valarray (const slice_array< _Tp > &)`
- `std::valarray< _Tp >::valarray (const valarray &)`
- `std::valarray< _Tp >::valarray (initializer_list< _Tp >)`
- `std::valarray< _Tp >::valarray (size_t)`
- `std::valarray< _Tp >::valarray (valarray &&) noexcept`
- `std::gslice::~gslice ()`
- `_Expr< _ValFunClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::apply (_Tp __func(_Tp)) const`
- `_Expr< _RefFunClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::apply (_Tp __func(const _Tp &)) const`
- `template<class _Tp >`
`const _Tp * std::begin (const valarray< _Tp > &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::begin (valarray< _Tp > &__va) noexcept`
- `valarray< _Tp > std::valarray< _Tp >::cshift (int __n) const`
- `template<class _Tp >`
`const _Tp * std::end (const valarray< _Tp > &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::end (valarray< _Tp > &__va) noexcept`
- `_Tp std::valarray< _Tp >::max () const`
- `_Tp std::valarray< _Tp >::min () const`
- `template<class _Dom >`
`valarray< _Tp > & std::valarray< _Tp >::operator= (const Expr< _Dom, _Tp > &)`
- `template<class _Dom >`
`void std::gslice_array< _Tp >::operator= (const Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::indirect_array< _Tp >::operator= (const Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::slice_array< _Tp >::operator= (const Expr< _Dom, _Tp > &) const`
- `template<class _Ex >`
`void std::mask_array< _Tp >::operator= (const Expr< _Ex, _Tp > &__e) const`
- `void std::gslice_array< _Tp >::operator= (const _Tp &) const`
- `void std::indirect_array< _Tp >::operator= (const _Tp &) const`
- `void std::mask_array< _Tp >::operator= (const _Tp &) const`

- `void std::slice_array< _Tp >::operator= (const _Tp &) const`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const _Tp &__t)`
- `gslice & std::gslice::operator= (const gslice &)`
- `gslice_array & std::gslice_array< _Tp >::operator= (const gslice_array &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const gslice_array< _Tp > &__ga)`
- `indirect_array & std::indirect_array< _Tp >::operator= (const indirect_array &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const indirect_array< _Tp > &__ia)`
- `mask_array & std::mask_array< _Tp >::operator= (const mask_array &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const mask_array< _Tp > &__ma)`
- `slice_array & std::slice_array< _Tp >::operator= (const slice_array &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const slice_array< _Tp > &__sa)`
- `void std::gslice_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::indirect_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::mask_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::slice_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (const valarray< _Tp > &__v)`
- `valarray & std::valarray< _Tp >::operator= (initializer_list< _Tp > __l)`
- `valarray< _Tp > & std::valarray< _Tp >::operator= (valarray< _Tp > &&__v) noexcept`
- `gslice_array< _Tp > std::valarray< _Tp >::operator[] (const gslice &__s)`
- `_Expr< _GClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (const gslice &__s) const`
- `mask_array< _Tp > std::valarray< _Tp >::operator[] (const valarray< bool > &__m)`
- `valarray< _Tp > std::valarray< _Tp >::operator[] (const valarray< bool > &__m) const`
- `indirect_array< _Tp > std::valarray< _Tp >::operator[] (const valarray< size_t > &__i)`
- `_Expr< _IClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (const valarray< size_t > &__i) const`
- `_Tp & std::valarray< _Tp >::operator[] (size_t __i) noexcept`
- `const _Tp & std::valarray< _Tp >::operator[] (size_t) const noexcept`
- `slice_array< _Tp > std::valarray< _Tp >::operator[] (slice __s)`
- `_Expr< _SClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (slice __s) const`
- `void std::valarray< _Tp >::resize (size_t __size, _Tp __c= _Tp())`
- `valarray< _Tp > std::valarray< _Tp >::shift (int __n) const`
- `valarray< size_t > std::gslice::size () const`
- `size_t std::slice::size () const`
- `size_t std::valarray< _Tp >::size () const`
- `size_t std::gslice::start () const`
- `size_t std::slice::start () const`
- `valarray< size_t > std::gslice::stride () const`
- `size_t std::slice::stride () const`
- `_Tp std::valarray< _Tp >::sum () const`
- `void std::valarray< _Tp >::swap (valarray< _Tp > &__v) noexcept`
- `template<typename _Tp, size_t _Nm>`
`std::valarray (const _Tp(&)[_Nm], size_t) -> valarray< _Tp >`

3.11.7.1 Detailed Description

Classes and functions for representing and manipulating arrays of elements.

3.11.7.2 Function Documentation

`gslice()` [1/3]

```
std::gslice::gslice ( ) [inline]
```

Construct an empty slice.

gslice() [2/3]

```
std::gslice::gslice (
    const gslice & __g ) [inline]
```

Copy constructor.

gslice() [3/3]

```
std::gslice::gslice (
    size_t __o,
    const valarray< size_t > & __l,
    const valarray< size_t > & __s ) [inline]
```

Construct a slice.

Constructs a slice with as many dimensions as the length of the *l* and *s* arrays.

Parameters

<code>__o</code>	Offset in array of first element.
<code>__l</code>	Array of dimension lengths.
<code>__s</code>	Array of dimension strides between array elements.

gslice_array()

```
template<typename _Tp >
std::gslice_array< _Tp >::gslice_array (
    const gslice_array< _Tp > & __a ) [inline]
```

Copy constructor. Both slices refer to the same underlying array.

indirect_array()

```
template<typename _Tp >
std::indirect_array< _Tp >::indirect_array (
    const indirect_array< _Tp > & __a ) [inline]
```

Copy constructor. Both slices refer to the same underlying array.

mask_array()

```
template<typename _Tp >
std::mask_array< _Tp >::mask_array (
    const mask_array< _Tp > & __a ) [inline]
```

Copy constructor. Both slices refer to the same underlying array.

slice() [1/2]

```
std::slice::slice ( ) [inline]
```

Construct an empty slice.

slice() [2/2]

```
std::slice::slice (
    size_t __o,
```

```
    size_t __d,
    size_t __s ) [inline]
```

Construct a slice.

Parameters

<code>__o</code>	Offset in array of first element.
<code>__d</code>	Number of elements in slice.
<code>__s</code>	Stride between array elements.

`slice_array()`

```
template<typename _Tp >
std::slice_array< _Tp >::slice_array (
    const slice_array< _Tp > & __a ) [inline]
```

Copy constructor. Both slices refer to the same underlying array.

`valarray()` [1/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray [inline], [noexcept]
```

Construct an empty array.

`valarray()` [2/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const _Tp & __t,
    size_t __n ) [inline]
```

Construct an array with n elements initialized to t .

`valarray()` [3/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const gslice_array< _Tp > & __ga ) [inline]
```

Construct an array with the same size and values in ga .

`valarray()` [4/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const indirect_array< _Tp > & __ia ) [inline]
```

Construct an array with the same size and values in ia .

`valarray()` [5/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const mask_array< _Tp > & __ma ) [inline]
```

Construct an array with the same size and values in ma .

valarray() [6/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const slice_array< _Tp > & __sa ) [inline]
```

Construct an array with the same size and values in *sa*.

valarray() [7/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    const valarray< _Tp > & __v ) [inline]
```

Copy constructor.

valarray() [8/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    initializer_list< _Tp > __l ) [inline]
```

Construct an array with an *initializer_list* of values.

valarray() [9/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    size_t __n ) [inline], [explicit]
```

Construct an array with *n* elements.

valarray() [10/10]

```
template<typename _Tp >
std::valarray< _Tp >::valarray (
    valarray< _Tp > && __v ) [inline], [noexcept]
```

Move constructor.

~gslice()

```
std::gslice::~gslice ( ) [inline]
```

Destructor.

apply() [1/2]

```
template<class _Tp >
_Expr< _ValFuncClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::apply (
    _Tp __func_Tp ) const [inline]
```

Apply a function to the array.

Returns a new valarray with elements assigned to the result of applying *__func* to the corresponding element of this array. The new array has the same size as this one.

Parameters

<i>__func</i>	Function of <i>Tp</i> returning <i>Tp</i> to apply.
---------------	---

Returns

New valarray with transformed elements.

apply() [2/2]

```
template<class _Tp >
_Expr< _RefFuncClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::apply (
    _Tp __funcconst _Tp & ) const [inline]
```

Apply a function to the array.

Returns a new valarray with elements assigned to the result of applying __func to the corresponding element of this array. The new array has the same size as this one.

Parameters

<code>__func</code>	Function of const Tp& returning Tp to apply.
---------------------	--

Returns

New valarray with transformed elements.

begin() [1/2]

```
template<class _Tp >
const _Tp * std::begin (
    const valarray< _Tp > & __va ) [inline], [noexcept]
```

Return an iterator pointing to the first element of the const valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

begin() [2/2]

```
template<class _Tp >
_Tp * std::begin (
    valarray< _Tp > & __va ) [inline], [noexcept]
```

Return an iterator pointing to the first element of the valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Referenced by [std::cbegin\(\)](#), [std::vector< _Tp, _Alloc >::insert\(\)](#), [std::list< _Tp, _Alloc >::merge\(\)](#), [std::list< _Tp, _Alloc >::merge\(\)](#), [std::vector< _Tp, _Alloc >::operator=\(\)](#), [std::list< _Tp, _Alloc >::remove\(\)](#), [std::list< _Tp, _Alloc >::remove_if\(\)](#), [std::list< _Tp, _Alloc >::sort\(\)](#), [std::list< _Tp, _Alloc >::sort\(\)](#), [std::list< _Tp, _Alloc >::unique\(\)](#), and [std::list< _Tp, _Alloc >::unique\(\)](#).

cshift()

```
template<class _Tp >
valarray< _Tp > std::valarray< _Tp >::cshift (
    int __n ) const [inline]
```

Return a rotated array.

A new valarray is constructed as a copy of this array with elements in shifted positions. For an element with index i , the new position is $(i - n) \% \text{size}()$. The new valarray has the same size as the current one. Elements that are shifted beyond the array bounds are shifted into the other end of the array. No elements are lost.

Positive arguments shift toward index 0, wrapping around the top. Negative arguments shift towards the top, wrapping around to 0.

Parameters

\leftarrow	Number of element positions to rotate.
n	

Returns

New valarray with elements in shifted positions.

end() [1/2]

```
template<class _Tp >
const _Tp * std::end (
    const valarray< _Tp > & __va ) [inline], [noexcept]
```

Return an iterator pointing to one past the last element of the const valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

end() [2/2]

```
template<class _Tp >
_Tp * std::end (
    valarray< _Tp > & __va ) [inline], [noexcept]
```

Return an iterator pointing to one past the last element of the valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Referenced by `std::cend()`, `std::vector< _Tp, _Alloc >::insert()`, `std::list< _Tp, _Alloc >::merge()`, `std::list< _Tp, _Alloc >::merge()`, `std::vector< _State< _TraitsT::char_type > >::operator=()`, `std::vector< _Tp, _Alloc >::operator=()`, `std::list< _Tp, _Alloc >::remove()`, `std::list< _Tp, _Alloc >::remove_if()`, `std::list< _Tp, _Alloc >::resize()`, `std::list< _Tp, _Alloc >::resize()`, `std::list< _Tp, _Alloc >::sort()`, `std::list< _Tp, _Alloc >::sort()`, `std::list< _Tp, _Alloc >::unique()`, and `std::list< _Tp, _Alloc >::unique()`.

max()

```
template<typename _Tp >
_Tp std::valarray< _Tp >::max [inline]
Return the maximum element using operator<().
```

min()

```
template<typename _Tp >
```

```
_Tp std::valarray<_Tp>::min [inline]
```

Return the minimum element using operator<().

operator=() [1/20]

```
template<typename _Tp >
void std::gslice_array<_Tp>::operator= (
    const _Tp & __t ) const [inline]
```

Assign all slice elements to *t*.

operator=() [2/20]

```
template<typename _Tp >
void std::indirect_array<_Tp>::operator= (
    const _Tp & __t ) const [inline]
```

Assign all slice elements to *t*.

operator=() [3/20]

```
template<typename _Tp >
void std::mask_array<_Tp>::operator= (
    const _Tp & __t ) const [inline]
```

Assign all slice elements to *t*.

operator=() [4/20]

```
template<typename _Tp >
void std::slice_array<_Tp>::operator= (
    const _Tp & __t ) const [inline]
```

Assign all slice elements to *t*.

operator=() [5/20]

```
template<typename _Tp >
valarray<_Tp> & std::valarray<_Tp>::operator= (
    const _Tp & __t ) [inline]
```

Assign elements to a value.

Assign all elements of array to *t*.

Parameters

↩	Value for elements.
↩	
↩	
↩	
<i>t</i>	

operator=() [6/20]

```
gslice & std::gslice::operator= (
    const gslice & __g ) [inline]
```

Assignment operator.

operator=() [7/20]

```
template<typename _Tp >
gslice_array< _Tp > & std::gslice_array< _Tp >::operator= (
    const gslice_array< _Tp > & __a ) [inline]
```

Assignment operator. Assigns slice elements to corresponding elements of *a*.

operator=() [8/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    const gslice_array< _Tp > & __ga ) [inline]
```

Assign elements to an array subset.

Assign elements of array to values in *ga*. Results are undefined if *ga* does not have the same size as this array.

Parameters

<code>__ga</code>	Array slice to get values from.
-------------------	---------------------------------

operator=() [9/20]

```
template<typename _Tp >
indirect_array< _Tp > & std::indirect_array< _Tp >::operator= (
    const indirect_array< _Tp > & __a ) [inline]
```

Assignment operator. Assigns elements to corresponding elements of *a*.

operator=() [10/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    const indirect_array< _Tp > & __ia ) [inline]
```

Assign elements to an array subset.

Assign elements of array to values in *ia*. Results are undefined if *ia* does not have the same size as this array.

Parameters

<code>__ia</code>	Array slice to get values from.
-------------------	---------------------------------

operator=() [11/20]

```
template<typename _Tp >
mask_array< _Tp > & std::mask_array< _Tp >::operator= (
    const mask_array< _Tp > & __a ) [inline]
```

Assignment operator. Assigns elements to corresponding elements of *a*.

operator=() [12/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    const mask_array< _Tp > & __ma ) [inline]
```

Assign elements to an array subset.

Assign elements of array to values in *ma*. Results are undefined if *ma* does not have the same size as this array.

Parameters

<code>__ma</code>	Array slice to get values from.
-------------------	---------------------------------

operator=() [13/20]

```
template<typename _Tp >
slice_array< _Tp > & std::slice_array< _Tp >::operator= (
    const slice_array< _Tp > & __a ) [inline]
```

Assignment operator. Assigns slice elements to corresponding elements of *a*.

operator=() [14/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    const slice_array< _Tp > & __sa ) [inline]
```

Assign elements to an array subset.

Assign elements of array to values in *sa*. Results are undefined if *sa* does not have the same size as this array.

Parameters

<code>__sa</code>	Array slice to get values from.
-------------------	---------------------------------

operator=() [15/20]

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator= (
    const valarray< _Tp > & __v ) const [inline]
```

Assign slice elements to corresponding elements of *v*.

References `std::valarray< _Tp >::size()`.

operator=() [16/20]

```
template<typename _Tp >
void std::indirect_array< _Tp >::operator= (
    const valarray< _Tp > & __v ) const [inline]
```

Assign slice elements to corresponding elements of *v*.

operator=() [17/20]

```
template<typename _Tp >
void std::slice_array< _Tp >::operator= (
    const valarray< _Tp > & __v ) const [inline]
```

Assign slice elements to corresponding elements of *v*.

operator=() [18/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    const valarray< _Tp > & __v ) [inline]
```

Assign elements to an array.

Assign elements of array to values in *v*.

Parameters

<code>_↔</code>	Valarray to get values from.
<code>_v</code>	

operator=() [19/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    initializer_list< _Tp > __l ) [inline]
```

Assign elements to an initializer_list.

Assign elements of array to values in __l. Results are undefined if __l does not have the same size as this array.

Parameters

<code>↔</code>	initializer_list to get values from.
<code>_↔</code>	
<code>↔</code>	
<code>_↔</code>	
<code>/</code>	

operator=() [20/20]

```
template<typename _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator= (
    valarray< _Tp > && __v ) [inline], [noexcept]
```

Move assign elements to an array.

Move assign elements of array to values in v.

Parameters

<code>_↔</code>	Valarray to get values from.
<code>_v</code>	

operator[]() [1/9]

```
template<typename _Tp >
gslice_array< _Tp > std::valarray< _Tp >::operator[] (
    const gslice & __s ) [inline]
```

Return a reference to an array subset.

Returns a new valarray containing the elements of the array indicated by the gslice argument. The new valarray has the same size as the input gslice.

See also

gslice.

Parameters

<code>_↔</code>	The source gslice.
<code>_s</code>	

Returns

New valarray containing elements in `__s`.

operator[]() [2/9]

```
template<typename _Tp >
_Expr< _GClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (
    const gslice & __s ) const [inline]
```

Return an array subset.

Returns a slice_array referencing the elements of the array indicated by the slice argument.

See also

gslice.

Parameters

<code>__s</code>	The source slice.
------------------	-------------------

Returns

Slice_array referencing elements indicated by `__s`.

operator[]() [3/9]

```
template<typename _Tp >
mask_array< _Tp > std::valarray< _Tp >::operator[] (
    const valarray< bool > & __m ) [inline]
```

Return a reference to an array subset.

Returns a new mask_array referencing the elements of the array indicated by the argument. The input is a valarray of bool which represents a bitmask indicating which elements are part of the subset. Elements of the array are part of the subset if the corresponding element of the argument is true.

Parameters

<code>__m</code>	The valarray bitmask.
------------------	-----------------------

Returns

New valarray containing elements indicated by `__m`.

operator[]() [4/9]

```
template<typename _Tp >
valarray< _Tp > std::valarray< _Tp >::operator[] (
    const valarray< bool > & __m ) const [inline]
```

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the argument. The input is a valarray of bool which represents a bitmask indicating which elements should be copied into the new valarray. Each element of the array is added to the return valarray if the corresponding element of the argument is true.

Parameters

\leftarrow __m	The valarray bitmask.
---------------------	-----------------------

Returns

New valarray containing elements indicated by __m.

operator[]() [5/9]

```
template<typename _Tp >
indirect_array< _Tp > std::valarray< _Tp >::operator[] (
    const valarray< size_t > & __i ) [inline]
```

Return a reference to an array subset.

Returns an indirect_array referencing the elements of the array indicated by the argument. The elements in the argument are interpreted as the indices of elements of this valarray to include in the subset. The returned indirect_array refers to these elements.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow <i>i</i>	The valarray element index list.
--	----------------------------------

Returns

Indirect_array referencing elements in __i.

operator[]() [6/9]

```
template<typename _Tp >
_Expr< _IClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (
    const valarray< size_t > & __i ) const [inline]
```

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the argument. The elements in the argument are interpreted as the indices of elements of this valarray to copy to the return valarray.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow <i>i</i>	The valarray element index list.
--	----------------------------------

Returns

New valarray containing elements in __s.

operator[]() [7/9]

```
template<typename _Tp >
_Tp & std::valarray< _Tp >::operator[] (
    size_t __i ) [inline], [noexcept]
```

Return a reference to the i'th array element.

Parameters

<code>↔</code>	Index of element to return.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>i</code>	

Returns

Reference to the i'th element.

operator[]() [8/9]

```
template<typename _Tp >
slice_array< _Tp > std::valarray< _Tp >::operator[] (
    slice __s ) [inline]
```

Return a reference to an array subset.

Returns a new valarray containing the elements of the array indicated by the slice argument. The new valarray has the same size as the input slice.

See also

slice.

Parameters

<code>↔</code>	The source slice.
<code>s</code>	

Returns

New valarray containing elements in `s`.

operator[]() [9/9]

```
template<typename _Tp >
_Expr< _SClos< _ValArray, _Tp >, _Tp > std::valarray< _Tp >::operator[] (
    slice __s ) const [inline]
```

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the slice argument. The new valarray has the same size as the input slice.

See also

slice.

Parameters

<code>__l</code>	The source slice.
<code>__s</code>	

Returns

New valarray containing elements in `__s`.

resize()

```
template<class _Tp >
void std::valarray< _Tp >::resize (
    size_t __size,
    _Tp __c = _Tp() ) [inline]
```

Resize array.

Resize this array to `size` and set all elements to `c`. All references and iterators are invalidated.

Parameters

<code>__size</code>	New array size.
<code>__c</code>	New value for all elements.

shift()

```
template<class _Tp >
valarray< _Tp > std::valarray< _Tp >::shift (
    int __n ) const [inline]
```

Return a shifted array.

A new valarray is constructed as a copy of this array with elements in shifted positions. For an element with index `i`, the new position is `i - n`. The new valarray has the same size as the current one. New elements without a value are set to 0. Elements whose new position is outside the bounds of the array are discarded.

Positive arguments shift toward index 0, discarding elements `[0, n)`. Negative arguments discard elements from the top of the array.

Parameters

<code>__n</code>	Number of element positions to shift.
-------------------------	---------------------------------------

Returns

New valarray with elements in shifted positions.

size() [1/3]

```
valarray< size_t > std::gslice::size ( ) const [inline]
```

Return array of sizes of slice dimensions.

size() [2/3]

```
size_t std::slice::size ( ) const [inline]
```

Return size of slice.

size() [3/3]

```
template<class _Tp >
size_t std::valarray< _Tp >::size [inline]
Return the number of elements in array.
Referenced by std::gslice_array< _Tp >::operator=().
```

start() [1/2]

```
size_t std::gslice::start ( ) const [inline]
Return array offset of first slice element.
```

start() [2/2]

```
size_t std::slice::start ( ) const [inline]
Return array offset of first slice element.
```

stride() [1/2]

```
valarray< size_t > std::gslice::stride ( ) const [inline]
Return array of array strides for each dimension.
```

stride() [2/2]

```
size_t std::slice::stride ( ) const [inline]
Return array stride of slice.
```

sum()

```
template<class _Tp >
_Tp std::valarray< _Tp >::sum [inline]
Return the sum of all elements in the array.
Accumulates the sum of all elements into a Tp using +=. The order of adding the elements is unspecified.
```

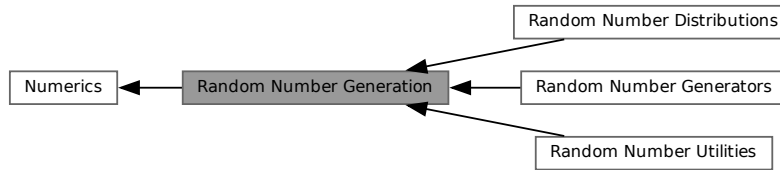
swap()

```
template<class _Tp >
void std::valarray< _Tp >::swap (
    valarray< _Tp > & __v ) [inline], [noexcept]
```

Swap.

3.11.8 Random Number Generation

Collaboration diagram for Random Number Generation:



Modules

- [Random Number Distributions](#)
- [Random Number Generators](#)
- [Random Number Utilities](#)

Functions

- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator > _RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`

3.11.8.1 Detailed Description

A facility for generating random numbers on selected distributions.

3.11.8.2 Function Documentation

generate_canonical()

```

template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >
_RealType std::generate_canonical (
    _UniformRandomNumberGenerator & __g )
  
```

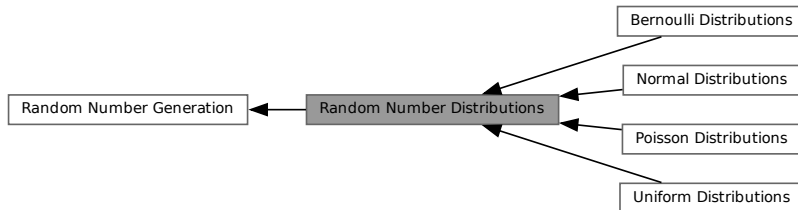
A function template for converting the output of a (integral) uniform random number generator to a floating point result in the range [0-1).

References [std::numeric_limits<_Tp>::epsilon\(\)](#), [std::generate_canonical\(\)](#), [std::log\(\)](#), and [std::min\(\)](#).

Referenced by [std::generate_canonical\(\)](#).

3.11.8.3 Random Number Distributions

Collaboration diagram for Random Number Distributions:



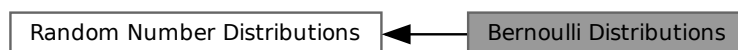
Modules

- [Bernoulli Distributions](#)
- [Normal Distributions](#)
- [Poisson Distributions](#)
- [Uniform Distributions](#)

3.11.8.3.1 Detailed Description

3.11.8.3.2 Bernoulli Distributions

Collaboration diagram for Bernoulli Distributions:



Classes

- class `std::bernoulli_distribution`
- class `std::binomial_distribution<_IntType>`
- class `std::geometric_distribution<_IntType>`
- class `std::negative_binomial_distribution<_IntType>`
- struct `std::bernoulli_distribution::param_type`
- struct `std::binomial_distribution<_IntType>::param_type`
- struct `std::geometric_distribution<_IntType>::param_type`
- struct `std::negative_binomial_distribution<_IntType>::param_type`

Functions

- bool `std::operator!=` (const `std::bernoulli_distribution` &__d1, const `std::bernoulli_distribution` &__d2)

- `template<typename _IntType >`
`bool std::operator!= (const std::binomial_distribution< _IntType > &__d1, const std::binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::geometric_distribution< _IntType > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::geometric_distribution< _IntType > &__x)`

Detailed Description

Function Documentation

operator"!=() [1/4]

```
bool std::operator!= (
    const std::bernoulli_distribution & __d1,
    const std::bernoulli_distribution & __d2 ) [inline]
```

Return true if two Bernoulli distributions have different parameters.

operator"!=() [2/4]

```
template<typename _IntType >
bool std::operator!= (
    const std::binomial_distribution< _IntType > & __d1,
    const std::binomial_distribution< _IntType > & __d2 ) [inline]
```

Return true if two binomial distributions are different.

operator"!=() [3/4]

```
template<typename _IntType >
bool std::operator!= (
    const std::geometric_distribution< _IntType > & __d1,
    const std::geometric_distribution< _IntType > & __d2 ) [inline]
```

Return true if two geometric distributions have different parameters.

operator"!=() [4/4]

```
template<typename _IntType >
bool std::operator!= (
    const std::negative_binomial_distribution< _IntType > & __d1,
    const std::negative_binomial_distribution< _IntType > & __d2 ) [inline]
```

Return true if two negative binomial distributions are different.

operator<<() [1/2]

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::bernoulli_distribution & __x )
```

Inserts a `bernoulli_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>bernoulli_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator<<() [2/2]

```
template<typename _IntType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::geometric_distribution< _IntType > & __x )
```

Inserts a `geometric_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>geometric_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

References `std::numeric_limits< _Tp >::epsilon()`, `std::log()`, and `std::numeric_limits< _Tp >::max()`.

operator>>() [1/2]

```
template<typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::bernoulli_distribution & __x ) [inline]
```

Extracts a `bernoulli_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>bernoulli_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

References [std::bernoulli_distribution::param\(\)](#).

operator>>() [2/2]

```
template<typename _IntType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::geometric_distribution< _IntType > & __x )
```

Extracts a `geometric_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>geometric_distribution</code> random number generator engine.

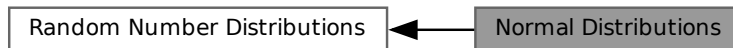
Returns

The input stream with `__x` extracted or in an error state.

References [std::ios_base::flags\(\)](#), and [std::geometric_distribution< _IntType >::param\(\)](#).

3.11.8.3.3 Normal Distributions

Collaboration diagram for Normal Distributions:

**Classes**

- class [std::cauchy_distribution< _RealType >](#)
- class [std::chi_squared_distribution< _RealType >](#)
- class [std::fisher_f_distribution< _RealType >](#)
- class [std::gamma_distribution< _RealType >](#)
- class [std::lognormal_distribution< _RealType >](#)
- class [std::normal_distribution< _RealType >](#)
- struct [std::normal_distribution< _RealType >::param_type](#)
- struct [std::lognormal_distribution< _RealType >::param_type](#)
- struct [std::gamma_distribution< _RealType >::param_type](#)
- struct [std::chi_squared_distribution< _RealType >::param_type](#)
- struct [std::cauchy_distribution< _RealType >::param_type](#)
- struct [std::fisher_f_distribution< _RealType >::param_type](#)
- struct [std::student_t_distribution< _RealType >::param_type](#)
- class [std::student_t_distribution< _RealType >](#)

Functions

- `template<typename _RealType >`
`bool std::operator!=(const std::cauchy_distribution< _RealType > &__d1, const std::cauchy_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::chi_squared_distribution< _RealType > &__d1, const std::chi_squared_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::fisher_f_distribution< _RealType > &__d1, const std::fisher_f_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::gamma_distribution< _RealType > &__d1, const std::gamma_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::lognormal_distribution< _RealType > &__d1, const std::lognormal_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::student_t_distribution< _RealType > &__d1, const std::student_t_distribution< _RealType > &__d2)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::cauchy_distribution< _RealType > &__x)`

Detailed Description

Function Documentation

`operator!=()` [1/7]

```
template<typename _RealType >
bool std::operator!=(
    const std::cauchy_distribution< _RealType > & __d1,
    const std::cauchy_distribution< _RealType > & __d2 ) [inline]
```

Return true if two Cauchy distributions have different parameters.

`operator!=()` [2/7]

```
template<typename _RealType >
bool std::operator!=(
    const std::chi_squared_distribution< _RealType > & __d1,
    const std::chi_squared_distribution< _RealType > & __d2 ) [inline]
```

Return true if two Chi-squared distributions are different.

`operator!=()` [3/7]

```
template<typename _RealType >
bool std::operator!=(
    const std::fisher_f_distribution< _RealType > & __d1,
    const std::fisher_f_distribution< _RealType > & __d2 ) [inline]
```

Return true if two Fisher f distributions are different.

operator"!=() [4/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::gamma_distribution< _RealType > & __d1,
    const std::gamma_distribution< _RealType > & __d2 ) [inline]
```

Return true if two gamma distributions are different.

operator"!=() [5/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::lognormal_distribution< _RealType > & __d1,
    const std::lognormal_distribution< _RealType > & __d2 ) [inline]
```

Return true if two lognormal distributions are different.

operator"!=() [6/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::normal_distribution< _RealType > & __d1,
    const std::normal_distribution< _RealType > & __d2 ) [inline]
```

Return true if two normal distributions are different.

operator"!=() [7/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::student_t_distribution< _RealType > & __d1,
    const std::student_t_distribution< _RealType > & __d2 ) [inline]
```

Return true if two Student t distributions are different.

operator<<()

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::cauchy_distribution< _RealType > & __x )
```

Inserts a cauchy_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A cauchy_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

References [std::tan\(\)](#).

operator>>()

```
template<typename _RealType , typename _CharT , typename _Traits >
```

```
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::cauchy_distribution< _RealType > & __x )
```

Extracts a `cauchy_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>cauchy_distribution</code> random number generator engine.

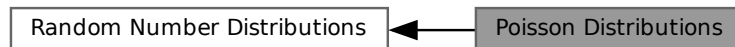
Returns

The input stream with `__x` extracted or in an error state.

References `std::ios_base::flags()`, and `std::cauchy_distribution< _RealType >::param()`.

3.11.8.3.4 Poisson Distributions

Collaboration diagram for Poisson Distributions:



Classes

- class `std::discrete_distribution< _IntType >`
- class `std::exponential_distribution< _RealType >`
- class `std::extreme_value_distribution< _RealType >`
- struct `std::poisson_distribution< _IntType >::param_type`
- struct `std::exponential_distribution< _RealType >::param_type`
- struct `std::weibull_distribution< _RealType >::param_type`
- struct `std::extreme_value_distribution< _RealType >::param_type`
- struct `std::discrete_distribution< _IntType >::param_type`
- struct `std::piecewise_constant_distribution< _RealType >::param_type`
- struct `std::piecewise_linear_distribution< _RealType >::param_type`
- class `std::piecewise_constant_distribution< _RealType >`
- class `std::piecewise_linear_distribution< _RealType >`
- class `std::poisson_distribution< _IntType >`
- class `std::weibull_distribution< _RealType >`

Functions

- template<typename `_IntType` >
 bool `std::operator!=` (const `std::discrete_distribution< _IntType >` & __d1, const `std::discrete_distribution< _IntType >` & __d2)

- `template<typename _RealType >`
`bool std::operator!=(const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!=(const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!=(const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _RealType > &__d2)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::weibull_distribution< _RealType > &__x)`

Detailed Description

Function Documentation

`operator"!="()` [1/7]

```
template<typename _IntType >
bool std::operator!=(
    const std::discrete_distribution< _IntType > & __d1,
    const std::discrete_distribution< _IntType > & __d2 ) [inline]
```

Return true if two discrete distributions have different parameters.

`operator"!="()` [2/7]

```
template<typename _RealType >
bool std::operator!=(
    const std::exponential_distribution< _RealType > & __d1,
    const std::exponential_distribution< _RealType > & __d2 ) [inline]
```

Return true if two exponential distributions have different parameters.

operator"!=() [3/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::extreme_value_distribution< _RealType > & __d1,
    const std::extreme_value_distribution< _RealType > & __d2 ) [inline]
```

Return true if two extreme value distributions have different parameters.

operator"!=() [4/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::piecewise_constant_distribution< _RealType > & __d1,
    const std::piecewise_constant_distribution< _RealType > & __d2 ) [inline]
```

Return true if two piecewise constant distributions have different parameters.

operator"!=() [5/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::piecewise_linear_distribution< _RealType > & __d1,
    const std::piecewise_linear_distribution< _RealType > & __d2 ) [inline]
```

Return true if two piecewise linear distributions have different parameters.

operator"!=() [6/7]

```
template<typename _IntType >
bool std::operator!= (
    const std::poisson_distribution< _IntType > & __d1,
    const std::poisson_distribution< _IntType > & __d2 ) [inline]
```

Return true if two Poisson distributions are different.

operator"!=() [7/7]

```
template<typename _RealType >
bool std::operator!= (
    const std::weibull_distribution< _RealType > & __d1,
    const std::weibull_distribution< _RealType > & __d2 ) [inline]
```

Return true if two Weibull distributions have different parameters.

operator<<() [1/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::exponential_distribution< _RealType > & __x )
```

Inserts a exponential_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A exponential_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

References [std::log\(\)](#).

operator<<() [2/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::extreme_value_distribution< _RealType > & __x )
```

Inserts a `extreme_value_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>extreme_value_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

References [std::log\(\)](#).

operator<<() [3/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::weibull_distribution< _RealType > & __x )
```

Inserts a `weibull_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>weibull_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

References [std::log\(\)](#), and [std::pow\(\)](#).

operator>>() [1/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::exponential_distribution< _RealType > & __x )
```

Extracts a `exponential_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
-------------------	------------------

Parameters

<code>__is</code>	A exponential_distribution random number generator engine.
-------------------	--

Returns

The input stream with `__x` extracted or in an error state.

References [std::ios_base::flags\(\)](#), and [std::exponential_distribution<_RealType>::param\(\)](#).

operator>>() [2/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::extreme_value_distribution< _RealType > & __x )
```

Extracts a extreme_value_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A extreme_value_distribution random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

References [std::ios_base::flags\(\)](#), and [std::extreme_value_distribution<_RealType>::param\(\)](#).

operator>>() [3/3]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::weibull_distribution< _RealType > & __x )
```

Extracts a weibull_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A weibull_distribution random number generator engine.

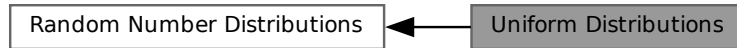
Returns

The input stream with `___x` extracted or in an error state.

References `std::ios_base::flags()`, and `std::weibull_distribution<_RealType>::param()`.

3.11.8.3.5 Uniform Distributions

Collaboration diagram for Uniform Distributions:

**Classes**

- struct `std::uniform_real_distribution<_RealType>::param_type`
- class `std::uniform_real_distribution<_RealType>`

Functions

- template<typename _IntType >
bool `std::operator!=` (const `std::uniform_int_distribution<_IntType>` &__d1, const `std::uniform_int_distribution<_IntType>` &__d2)
- template<typename _IntType >
bool `std::operator!=` (const `std::uniform_real_distribution<_IntType>` &__d1, const `std::uniform_real_distribution<_IntType>` &__d2)
- template<typename _IntType, typename _CharT, typename _Traits >
`std::basic_ostream<_CharT, _Traits>` & `std::operator<<` (`std::basic_ostream<_CharT, _Traits>` &, const `std::uniform_int_distribution<_IntType>` &)
- template<typename _RealType, typename _CharT, typename _Traits >
`std::basic_ostream<_CharT, _Traits>` & `std::operator<<` (`std::basic_ostream<_CharT, _Traits>` &, const `std::uniform_real_distribution<_RealType>` &)
- template<typename _IntType, typename _CharT, typename _Traits >
`std::basic_istream<_CharT, _Traits>` & `std::operator>>` (`std::basic_istream<_CharT, _Traits>` &, `std::uniform_int_distribution<_IntType>` &)
- template<typename _RealType, typename _CharT, typename _Traits >
`std::basic_istream<_CharT, _Traits>` & `std::operator>>` (`std::basic_istream<_CharT, _Traits>` &, `std::uniform_real_distribution<_RealType>` &)

Detailed Description**Function Documentation****`operator!=()` [1/2]**

```

template<typename _IntType >
bool std::operator!= (
    const std::uniform_int_distribution<_IntType> & __d1,
    const std::uniform_int_distribution<_IntType> & __d2 ) [inline]
  
```

Return true if two uniform integer distributions have different parameters.

operator!=() [2/2]

```
template<typename _IntType >
bool std::operator!= (
    const std::uniform_real_distribution< _IntType > & __d1,
    const std::uniform_real_distribution< _IntType > & __d2 ) [inline]
```

Return true if two uniform real distributions have different parameters.

operator<<() [1/2]

```
template<typename _IntType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::uniform_int_distribution< _IntType > & __x )
```

Inserts a uniform_int_distribution random number distribution __x into the output stream os.

Parameters

__os	An output stream.
__x	A uniform_int_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

References [std::ios_base::flags\(\)](#).

operator<<() [2/2]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::uniform_real_distribution< _RealType > & __x )
```

Inserts a uniform_real_distribution random number distribution __x into the output stream __os.

Parameters

__os	An output stream.
__x	A uniform_real_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

operator>>() [1/2]

```
template<typename _IntType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::uniform_int_distribution< _IntType > & __x )
```

Extracts a uniform_int_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>uniform_int_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

References [`std::ios_base::flags\(\)`](#), and [`std::uniform_int_distribution<_IntType>::param\(\)`](#).

operator>>() [2/2]

```
template<typename _RealType , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::uniform_real_distribution< _RealType > & __x )
```

Extracts a `uniform_real_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>uniform_real_distribution</code> random number generator engine.

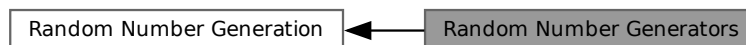
Returns

The input stream with `__x` extracted or in an error state.

References [`std::ios_base::flags\(\)`](#), and [`std::uniform_real_distribution<_RealType>::param\(\)`](#).

3.11.8.4 Random Number Generators

Collaboration diagram for Random Number Generators:

**Classes**

- class [`std::discard_block_engine<_RandomNumberEngine, __p, __r>`](#)
- class [`std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>`](#)
- class [`std::linear_congruential_engine<_UIntType, __a, __c, __m>`](#)
- class [`std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>`](#)
- class [`std::random_device`](#)

- class `std::shuffle_order_engine<_RandomNumberEngine, __k>`
- class `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>`

Typedefs

- typedef `minstd_rand0` `std::default_random_engine`
- typedef `shuffle_order_engine< minstd_rand0, 256 >` `std::knuth_b`
- typedef `linear_congruential_engine< uint_fast32_t, 48271UL, 0UL, 2147483647UL >` `std::minstd_rand`
- typedef `linear_congruential_engine< uint_fast32_t, 16807UL, 0UL, 2147483647UL >` `std::minstd_rand0`
- typedef `mersenne_twister_engine< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL >` `std::mt19937`
- typedef `mersenne_twister_engine< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL >` `std::mt19937_64`
- typedef `discard_block_engine< ranlux24_base, 223, 23 >` `std::ranlux24`
- typedef `subtract_with_carry_engine< uint_fast32_t, 24, 10, 24 >` `std::ranlux24_base`
- typedef `discard_block_engine< ranlux48_base, 389, 11 >` `std::ranlux48`
- typedef `subtract_with_carry_engine< uint_fast64_t, 48, 5, 12 >` `std::ranlux48_base`

Functions

- template<typename _RandomNumberEngine, size_t __p, size_t __r>
bool `std::operator!=` (const `std::discard_block_engine<_RandomNumberEngine, __p, __r>` &__lhs, const `std::discard_block_engine<_RandomNumberEngine, __p, __r>` &__rhs)
- template<typename _RandomNumberEngine, size_t __w, typename _UIntType >
bool `std::operator!=` (const `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>` &__lhs, const `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>` &__rhs)
- template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>
bool `std::operator!=` (const `std::linear_congruential_engine<_UIntType, __a, __c, __m>` &__lhs, const `std::linear_congruential_engine<_UIntType, __a, __c, __m>` &__rhs)
- template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
bool `std::operator!=` (const `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>` &__lhs, const `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>` &__rhs)
- template<typename _RandomNumberEngine, size_t __k>
bool `std::operator!=` (const `std::shuffle_order_engine<_RandomNumberEngine, __k>` &__lhs, const `std::shuffle_order_engine<_RandomNumberEngine, __k>` &__rhs)
- template<typename _UIntType, size_t __w, size_t __s, size_t __r>
bool `std::operator!=` (const `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>` &__lhs, const `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>` &__rhs)
- template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >
`std::basic_ostream<_CharT, _Traits>` & `std::operator<<` (`std::basic_ostream<_CharT, _Traits>` &__os, const `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>` &__x)

3.11.8.4.1 Detailed Description

These classes define objects which provide random or pseudorandom numbers, either from a discrete or a continuous interval. The random number generator supplied as a part of this library are all uniform random number generators which provide a sequence of random number uniformly distributed over their range.

A number generator is a function object with an operator() that takes zero arguments and returns a number.

A compliant random number generator must satisfy the following requirements.

Table 292 Random Number Generator Requirements

To be documented.

3.11.8.4.2 Typedef Documentation

minstd_rand

```
typedef linear_congruential_engine<uint_fast32_t, 48271UL, 0UL, 2147483647UL> std::minstd_rand
```

An alternative LCR (Lehmer Generator function).

minstd_rand0

```
typedef linear_congruential_engine<uint_fast32_t, 16807UL, 0UL, 2147483647UL> std::minstd_rand0
```

The classic Minimum Standard rand0 of Lewis, Goodman, and Miller.

mt19937

```
typedef mersenne_twister_engine< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL,
7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL> std::mt19937
```

The classic Mersenne Twister.

Reference: M. Matsumoto and T. Nishimura, Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator, ACM Transactions on Modeling and Computer Simulation, Vol. 8, No. 1, January 1998, pp 3-30.

mt19937_64

```
typedef mersenne_twister_engine< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29,
0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL>
std::mt19937_64
```

An alternative Mersenne Twister.

3.11.8.4.3 Function Documentation

operator"!=() [1/6]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
bool std::operator!=(
    const std::discard_block_engine< _RandomNumberEngine, __p, __r > & __lhs,
    const std::discard_block_engine< _RandomNumberEngine, __p, __r > & __rhs ) [inline]
```

Compares two discard_block_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A discard_block_engine random number generator object.
<code>__rhs</code>	Another discard_block_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator"!=() [2/6]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
bool std::operator!=(
```

```

const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __lhs,
const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __rhs )
[inline]

```

Compares two independent_bits_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A independent_bits_engine random number generator object.
<code>__rhs</code>	Another independent_bits_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator"!=() [3/6]

```

template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
bool std::operator!= (
    const std::linear_congruential_engine< _UIntType, __a, __c, __m > & __lhs,
    const std::linear_congruential_engine< _UIntType, __a, __c, __m > & __rhs ) [inline]

```

Compares two linear congruential random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A linear congruential random number generator object.
<code>__rhs</code>	Another linear congruential random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator"!=() [4/6]

```

template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
bool std::operator!= (
    const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __↵
__s, __b, __t, __c, __l, __f > & __lhs,
    const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __↵
__s, __b, __t, __c, __l, __f > & __rhs ) [inline]

```

Compares two % mersenne_twister_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A % mersenne_twister_engine random number generator object.
<code>__rhs</code>	Another % mersenne_twister_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator"!="() [5/6]

```
template<typename _RandomNumberEngine , size_t __k>
bool std::operator!= (
    const std::shuffle_order_engine< _RandomNumberEngine, __k > & __lhs,
    const std::shuffle_order_engine< _RandomNumberEngine, __k > & __rhs ) [inline]
```

Compares two shuffle_order_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A shuffle_order_engine random number generator object.
<code>__rhs</code>	Another shuffle_order_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator"!="() [6/6]

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
bool std::operator!= (
    const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > & __lhs,
    const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > & __rhs ) [inline]
```

Compares two % subtract_with_carry_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A % subtract_with_carry_engine random number generator object.
<code>__rhs</code>	Another % subtract_with_carry_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

operator<<()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType , typename _CharT , typename
_Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __x )
```

Inserts the current state of a independent_bits_engine random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A independent_bits_engine random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

3.11.8.5 Random Number Utilities

Collaboration diagram for Random Number Utilities:



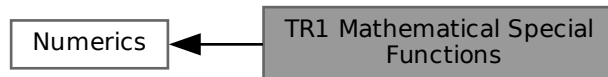
Classes

- class `std::seed_seq`

3.11.8.5.1 Detailed Description

3.11.9 TR1 Mathematical Special Functions

Collaboration diagram for TR1 Mathematical Special Functions:



Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_laguerre` (unsigned int __n, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_laguerref` (unsigned int __n, unsigned int __m, float __x)
- `long double std::tr1::assoc_laguerrel` (unsigned int __n, unsigned int __m, long double __x)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_legendre` (unsigned int __l, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_legendref` (unsigned int __l, unsigned int __m, float __x)
- `long double std::tr1::assoc_legendrel` (unsigned int __l, unsigned int __m, long double __x)
- `template<typename _Tpx, typename _Tpy >`
`__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type std::tr1::beta` (_Tpx __x, _Tpy __y)
- `float std::tr1::betaf` (float __x, float __y)
- `long double std::tr1::betal` (long double __x, long double __y)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_1` (_Tp __k)
- `float std::tr1::comp_ellint_1f` (float __k)

- long double **std::tr1::comp_ellint_1l** (long double __k)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::comp_ellint_2** (_Tp __k)
- float **std::tr1::comp_ellint_2f** (float __k)
- long double **std::tr1::comp_ellint_2l** (long double __k)
- template<typename _Tp, typename _Tpn >
__gnu_cxx::__promote_2< _Tp, _Tpn >::__type **std::tr1::comp_ellint_3** (_Tp __k, _Tpn __nu)
- float **std::tr1::comp_ellint_3f** (float __k, float __nu)
- long double **std::tr1::comp_ellint_3l** (long double __k, long double __nu)
- template<typename _Tpa, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type **std::tr1::conf_hyperg** (_Tpa __a, _Tpc __c, _Tp __x)
- float **std::tr1::conf_hypergf** (float __a, float __c, float __x)
- long double **std::tr1::conf_hypergl** (long double __a, long double __c, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_bessel_i** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_bessel_if** (float __nu, float __x)
- long double **std::tr1::cyl_bessel_il** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_bessel_j** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_bessel_jf** (float __nu, float __x)
- long double **std::tr1::cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_bessel_kf** (float __nu, float __x)
- long double **std::tr1::cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_neumannf** (float __nu, float __x)
- long double **std::tr1::cyl_neumannl** (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **std::tr1::ellint_1** (_Tp __k, _Tpp __phi)
- float **std::tr1::ellint_1f** (float __k, float __phi)
- long double **std::tr1::ellint_1l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **std::tr1::ellint_2** (_Tp __k, _Tpp __phi)
- float **std::tr1::ellint_2f** (float __k, float __phi)
- long double **std::tr1::ellint_2l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type **std::tr1::ellint_3** (_Tp __k, _Tpn __nu, _Tpp __phi)
- float **std::tr1::ellint_3f** (float __k, float __nu, float __phi)
- long double **std::tr1::ellint_3l** (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::expint** (_Tp __x)
- float **std::tr1::expintf** (float __x)
- long double **std::tr1::expintl** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::hermite** (unsigned int __n, _Tp __x)
- float **std::tr1::hermitef** (unsigned int __n, float __x)
- long double **std::tr1::hermitel** (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type **std::tr1::hyperg** (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)

- float **std::tr1::hypergf** (float __a, float __b, float __c, float __x)
- long double **std::tr1::hypergl** (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::laguerre** (unsigned int __n, _Tp __x)
- float **std::tr1::laguerref** (unsigned int __n, float __x)
- long double **std::tr1::laguerrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::legendre** (unsigned int __n, _Tp __x)
- float **std::tr1::legendref** (unsigned int __n, float __x)
- long double **std::tr1::legendrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::riemann_zeta** (_Tp __x)
- float **std::tr1::riemann_zetaf** (float __x)
- long double **std::tr1::riemann_zetal** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_bessel** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_besself** (unsigned int __n, float __x)
- long double **std::tr1::sph_bessell** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_legendre** (unsigned int __l, unsigned int __m, _Tp __theta)
- float **std::tr1::sph_legendref** (unsigned int __l, unsigned int __m, float __theta)
- long double **std::tr1::sph_legendrel** (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_neumann** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_neumannf** (unsigned int __n, float __x)
- long double **std::tr1::sph_neumannl** (unsigned int __n, long double __x)

3.11.9.1 Detailed Description

A collection of advanced mathematical special functions.

3.11.9.2 Function Documentation

assoc_laguerre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_laguerre (
    unsigned int __n,
    unsigned int __m,
    _Tp __x ) [inline]
```

5.2.1.1 Associated Laguerre polynomials.

assoc_legendre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_legendre (
    unsigned int __l,
    unsigned int __m,
    _Tp __x ) [inline]
```

5.2.1.2 Associated Legendre functions.

beta()

```
template<typename _Tpx , typename _Tpy >
__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type std::tr1::beta (
    _Tpx __x,
    _Tpy __y ) [inline]
```

5.2.1.3 Beta functions.**comp_ellint_1()**

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_1 (
    _Tp __k ) [inline]
```

5.2.1.4 Complete elliptic integrals of the first kind.**comp_ellint_2()**

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_2 (
    _Tp __k ) [inline]
```

5.2.1.5 Complete elliptic integrals of the second kind.**comp_ellint_3()**

```
template<typename _Tp , typename _Tpn >
__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::tr1::comp_ellint_3 (
    _Tp __k,
    _Tpn __nu ) [inline]
```

5.2.1.6 Complete elliptic integrals of the third kind.**conf_hyperg()**

```
template<typename _Tpa , typename _Tpc , typename _Tp >
__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type std::tr1::conf_hyperg (
    _Tpa __a,
    _Tpc __c,
    _Tp __x ) [inline]
```

5.2.1.7 Confluent hypergeometric functions.**cyl_bessel_i()**

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_i (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

5.2.1.8 Regular modified cylindrical Bessel functions.**cyl_bessel_j()**

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_j (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

5.2.1.9 Cylindrical Bessel functions (of the first kind).

cyl_bessel_k()

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_k (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

5.2.1.10 Irregular modified cylindrical Bessel functions.

cyl_neumann()

```
template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_neumann (
    _Tpnu __nu,
    _Tp __x ) [inline]
```

5.2.1.11 Cylindrical Neumann functions.

ellint_1()

```
template<typename _Tp , typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_1 (
    _Tp __k,
    _Tpp __phi ) [inline]
```

5.2.1.12 Incomplete elliptic integrals of the first kind.

ellint_2()

```
template<typename _Tp , typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_2 (
    _Tp __k,
    _Tpp __phi ) [inline]
```

5.2.1.13 Incomplete elliptic integrals of the second kind.

ellint_3()

```
template<typename _Tp , typename _Tpn , typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type std::tr1::ellint_3 (
    _Tp __k,
    _Tpn __nu,
    _Tpp __phi ) [inline]
```

5.2.1.14 Incomplete elliptic integrals of the third kind.

expint()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::expint (
    _Tp __x ) [inline]
```

5.2.1.15 Exponential integrals.

hermite()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::tr1::hermite (
    unsigned int __n,
    _Tp __x ) [inline]
```

5.2.1.16 Hermite polynomials.

hyperg()

```
template<typename _Tpa , typename _Tpb , typename _Tpc , typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type std::trl::hyperg (
    _Tpa __a,
    _Tpb __b,
    _Tpc __c,
    _Tp __x ) [inline]
```

5.2.1.17 Hypergeometric functions.

laguerre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::laguerre (
    unsigned int __n,
    _Tp __x ) [inline]
```

5.2.1.18 Laguerre polynomials.

legendre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::legendre (
    unsigned int __n,
    _Tp __x ) [inline]
```

5.2.1.19 Legendre polynomials.

riemann_zeta()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::riemann_zeta (
    _Tp __x ) [inline]
```

5.2.1.20 Riemann zeta function.

sph_bessel()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::sph_bessel (
    unsigned int __n,
    _Tp __x ) [inline]
```

5.2.1.21 Spherical Bessel functions.

sph_legendre()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::sph_legendre (
    unsigned int __l,
    unsigned int __m,
    _Tp __theta ) [inline]
```

5.2.1.22 Spherical associated Legendre functions.

sph_neumann()

```
template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type std::trl::sph_neumann (
    unsigned int __n,
    _Tp __x ) [inline]
```

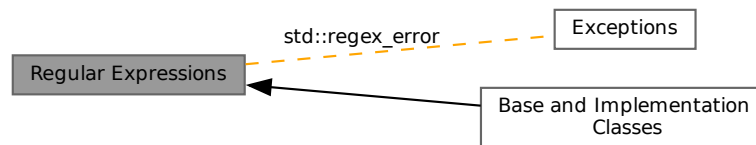
5.2.1.23 Spherical Neumann functions.

3.12 Ranges

Components for dealing with ranges of elements.

3.13 Regular Expressions

Collaboration diagram for Regular Expressions:



Modules

- [Base and Implementation Classes](#)

Namespaces

- namespace [std::regex_constants](#)

Classes

- class [std::basic_regex<_Ch_type, _Rx_traits>](#)
- class [std::match_results<_Bi_iter, _Alloc>](#)
- class [std::regex_error](#)
- class [std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits>](#)
- class [std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>](#)
- class [std::regex_traits<_Ch_type>](#)
- class [std::sub_match<_Biter>](#)

Typedefs

- typedef [match_results<const char*>](#) **std::cmatch**
- typedef [regex_iterator<const char*>](#) **std::cregex_iterator**
- typedef [regex_token_iterator<const char*>](#) **std::cregex_token_iterator**
- typedef [sub_match<const char*>](#) **std::csub_match**
- typedef [basic_regex<char>](#) **std::regex**
- typedef [match_results<string::const_iterator>](#) **std::smatch**
- typedef [regex_iterator<string::const_iterator>](#) **std::sregex_iterator**
- typedef [regex_token_iterator<string::const_iterator>](#) **std::sregex_token_iterator**
- typedef [sub_match<string::const_iterator>](#) **std::ssub_match**
- typedef [match_results<const wchar_t*>](#) **std::wcmatch**
- typedef [regex_iterator<const wchar_t*>](#) **std::wcregex_iterator**
- typedef [regex_token_iterator<const wchar_t*>](#) **std::wcregex_token_iterator**

- typedef `sub_match`< const wchar_t * > `std::wsub_match`
- typedef `basic_regex`< wchar_t > `std::wregex`
- typedef `match_results`< wstring::const_iterator > `std::wsmatch`
- typedef `regex_iterator`< wstring::const_iterator > `std::wsregex_iterator`
- typedef `regex_token_iterator`< wstring::const_iterator > `std::wsregex_token_iterator`
- typedef `sub_match`< wstring::const_iterator > `std::wssub_match`

Functions

- template<typename _ForwardIterator >
std::basic_regex (_ForwardIterator, _ForwardIterator, `regex_constants::syntax_option_type`={}) -> `basic_regex`< typename `iterator_traits`< _ForwardIterator >::value_type >
- template<typename _Bi_iter, class _Alloc >
bool **std::operator!=** (const `match_results`< _Bi_iter, _Alloc > &__m1, const `match_results`< _Bi_iter, _Alloc > &__m2)
- template<typename _Bi_iter, typename _Alloc >
bool **std::operator==** (const `match_results`< _Bi_iter, _Alloc > &__m1, const `match_results`< _Bi_iter, _Alloc > &__m2)
- template<typename _Ch_type, typename _Rx_traits >
void **swap** (`basic_regex`< _Ch_type, _Rx_traits > &__lhs, `basic_regex`< _Ch_type, _Rx_traits > &__rhs) noexcept
- template<typename _Bi_iter, typename _Alloc >
void **std::swap** (`match_results`< _Bi_iter, _Alloc > &__lhs, `match_results`< _Bi_iter, _Alloc > &__rhs) noexcept

Matching, Searching, and Replacing

- template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >
bool **std::regex_match** (_Bi_iter __s, _Bi_iter __e, `match_results`< _Bi_iter, _Alloc > &__m, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __flags=`regex_constants::match_default`)
- template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >
bool **std::regex_match** (_Bi_iter __first, _Bi_iter __last, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __flags=`regex_constants::match_default`)
- template<typename _Ch_type, typename _Alloc, typename _Rx_traits >
bool **std::regex_match** (const _Ch_type * __s, `match_results`< const _Ch_type *, _Alloc > &__m, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __f=`regex_constants::match_default`)
- template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
bool **std::regex_match** (const `basic_string`< _Ch_type, _Ch_traits, _Ch_alloc > &__s, `match_results`< typename `basic_string`< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __flags=`regex_constants::match_default`)
- template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
bool **std::regex_match** (const `basic_string`< _Ch_type, _Ch_traits, _Ch_alloc > &&, `match_results`< typename `basic_string`< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const `basic_regex`< _Ch_type, _Rx_traits > &, `regex_constants::match_flag_type`=`regex_constants::match_default`)=delete
- template<typename _Ch_type, class _Rx_traits >
bool **std::regex_match** (const _Ch_type * __s, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __f=`regex_constants::match_default`)
- template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >
bool **std::regex_match** (const `basic_string`< _Ch_type, _Ch_traits, _Str_allocator > &__s, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __flags=`regex_constants::match_default`)
- template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >
bool **std::regex_search** (_Bi_iter __s, _Bi_iter __e, `match_results`< _Bi_iter, _Alloc > &__m, const `basic_regex`< _Ch_type, _Rx_traits > &__re, `regex_constants::match_flag_type` __flags=`regex_constants::match_default`)

- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__e,`
`regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,`
`_Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const _Ch_type *__fmt, size_t __len, regex_constants::match_flag_type __flags)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type __f,`
`__flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, __`
`_Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s,`
`const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _Fst, _Fsa > &__fmt,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __f,`
`__flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx`
`_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type __f,`
`__flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits`
`> &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bilter >`
`bool operator== (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bilter >`
`bool operator!= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bilter >`
`bool operator< (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bilter >`
`bool operator<= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`

- [illegible]

- `template<typename _Bi_iter >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter >`
`basic_ostream< _Ch_type, _Ch_traits > & operator<< (basic_ostream< _Ch_type, _Ch_traits > &__os, const sub_match< _Bi_iter > &__m)`

3.13.1 Detailed Description

A facility for performing regular expression pattern matching.

3.13.2 Typedef Documentation

cregex_token_iterator

`typedef regex_token_iterator<const char*> std::cregex_token_iterator`
Token iterator for C-style NULL-terminated strings.

csub_match

`typedef sub_match<const char*> std::csub_match`
Standard regex submatch over a C-style null-terminated string.

regex

`typedef basic_regex<char> std::regex`
Standard regular expressions.

sregex_token_iterator

`typedef regex_token_iterator<string::const_iterator> std::sregex_token_iterator`
Token iterator for standard strings.

ssub_match

`typedef sub_match<string::const_iterator> std::ssub_match`
Standard regex submatch over a standard string.

wcregex_token_iterator

`typedef regex_token_iterator<const wchar_t*> std::wcregex_token_iterator`
Token iterator for C-style NULL-terminated wide strings.

wcsub_match

`typedef sub_match<const wchar_t*> std::wcsub_match`
Regex submatch over a C-style null-terminated wide string.

wregex

`typedef basic_regex<wchar_t> std::wregex`
Standard wide-character regular expressions.

wsregex_token_iterator

typedef [regex_token_iterator](#)<wstring::const_iterator> [std::wsregex_token_iterator](#)
 Token iterator for standard wide-character strings.

wssub_match

typedef [sub_match](#)<wstring::const_iterator> [std::wssub_match](#)
 Regex submatch over a standard wide string.

3.13.3 Function Documentation**__regex_replace()**

```
template<typename _Out_iter , typename _Bi_iter , typename _Rx_traits , typename _Ch_type >
_Out_iter std::__regex_replace (
    _Out_iter __out,
    _Bi_iter __first,
    _Bi_iter __last,
    const basic\_regex< _Ch_type, _Rx_traits > & __e,
    const _Ch_type * __fmt,
    size_t __len,
    regex\_constants::match\_flag\_type __flags )
```

Determines if there is a match between the regular expression *e* and all of the character sequence [*first*, *last*).

Parameters

<i>__s</i>	Start of the character sequence to match.
<i>__e</i>	One-past-the-end of the character sequence to match.
<i>__m</i>	The match results.
<i>__re</i>	The regular expression.
<i>__flags</i>	Controls how the regular expression is matched.

Return values

<i>true</i>	A match exists.
<i>false</i>	Otherwise.

Exceptions

<i>an</i>	exception of type regex_error .
-----------	---

Referenced by [std::regex_replace\(\)](#), and [std::regex_replace\(\)](#).

operator"!=() [1/8]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator!=(
    const sub\_match\_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub\_match< _Bi_iter > & __rhs ) [related]
```

Tests the inequivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator"!="() [2/8]

```
template<typename _Bi_iter , class _Alloc >
bool std::operator!= (
    const match_results< _Bi_iter, _Alloc > & __m1,
    const match_results< _Bi_iter, _Alloc > & __m2 ) [inline]
```

Compares two `match_results` for inequality.

Returns

true if the two objects do not refer to the same match, false otherwise.

operator"!="() [3/8]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator!= (
    const sub_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the inequivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator"!="() [4/8]

```
template<typename _Bi_iter >
bool operator!= (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the inequivalence of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator"!=([5/8]

```
template<typename _Bi_iter >
bool operator!= (
    const sub\_match< _Bi_iter > & __lhs,
    typename iterator\_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the inequivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator"!=([6/8]

```
template<typename _BiIter >
bool operator!= (
    const sub\_match< _BiIter > & __lhs,
    const sub\_match< _BiIter > & __rhs ) [related]
```

Tests the inequivalence of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

References [std::sub_match<_BiIter>::compare\(\)](#).

operator"!=([7/8]

```
template<typename _Bi_iter >
bool operator!= (
    typename iterator\_traits< _Bi_iter >::value_type const & __lhs,
    const sub\_match< _Bi_iter > & __rhs ) [related]
```

Tests the inequivalence of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator"!=([8/8]

```
template<typename _Bi_iter >
bool operator!= (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the inequivalence of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

operator<([1/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator< (
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

operator<([2/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator< (
    const sub_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

operator<() [3/7]

```
template<typename _Bi_iter >
bool operator< (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

References [std::__addressof\(\)](#).

operator<() [4/7]

```
template<typename _Bi_iter >
bool operator< (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a C string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

References [std::sub_match<_Biter>::compare\(\)](#).

operator<() [5/7]

```
template<typename _BiIter >
bool operator< (
    const sub_match< _BiIter > & __lhs,
    const sub_match< _BiIter > & __rhs ) [related]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

References [std::sub_match<_Biter>::compare\(\)](#).

operator<() [6/7]

```
template<typename _Bi_iter >
bool operator< (
    typename iterator_traits< _Bi_iter >::value_type const & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

References [std::__addressof\(\)](#).

operator<() [7/7]

```
template<typename _Bi_iter >
bool operator< (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

References [std::sub_match<_Biter>::compare\(\)](#).

operator<<()

```
template<typename _Ch_type , typename _Ch_traits , typename _Bi_iter >
basic_ostream< _Ch_type, _Ch_traits > & operator<< (
    basic_ostream< _Ch_type, _Ch_traits > & __os,
    const sub_match< _Bi_iter > & __m ) [related]
```

Inserts a matched string into an output stream.

Parameters

<code>__os</code>	The output stream.
<code>__m</code>	A submatch string.

Returns

the output stream with the submatch string inserted.

operator<=() [1/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator<= (
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator<=() [2/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator<= (
    const sub_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator<=() [3/7]

```
template<typename _Bi_iter >
bool operator<= (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator<=() [4/7]

```
template<typename _Bi_iter >
bool operator<= (
    const sub\_match< _Bi_iter > & __lhs,
    typename iterator\_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a C string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator<=() [5/7]

```
template<typename _BiIter >
bool operator<= (
    const sub\_match< _BiIter > & __lhs,
    const sub\_match< _BiIter > & __rhs ) [related]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

References [std::sub_match<_BiIter>::compare\(\)](#).

operator<=() [6/7]

```
template<typename _Bi_iter >
bool operator<= (
    typename iterator\_traits< _Bi_iter >::value_type const & __lhs,
    const sub\_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator<=() [7/7]

```
template<typename _Bi_iter >
bool operator<= (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

operator==([1/8]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator==(
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the equivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

operator==([2/8]

```
template<typename _Bi_iter , typename _Alloc >
bool std::operator==(
    const match_results< _Bi_iter, _Alloc > & __m1,
    const match_results< _Bi_iter, _Alloc > & __m2 ) [inline]
```

Compares two `match_results` for equality.

Returns

true if the two objects refer to the same match, false otherwise.

References `std::match_results< _Bi_iter, _Alloc >::begin()`, `std::match_results< _Bi_iter, _Alloc >::empty()`, `std::match_results< _Bi_iter, _Alloc >::prefix()`, `std::match_results< _Bi_iter, _Alloc >::ready()`, `std::match_results< _Bi_iter, _Alloc >::s` and `std::match_results< _Bi_iter, _Alloc >::suffix()`.

operator==() [3/8]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator== (
    const sub_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the equivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

operator==() [4/8]

```
template<typename _Bi_iter >
bool operator== (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the equivalence of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

References [std::__addressof\(\)](#).

operator==() [5/8]

```
template<typename _Bi_iter >
bool operator== (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the equivalence of a regular expression submatch and a C string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

References [std::sub_match< _Biter >::compare\(\)](#).

operator==() [6/8]

```
template<typename _BiIter >
bool operator== (
    const sub_match< _BiIter > & __lhs,
    const sub_match< _BiIter > & __rhs ) [related]
```

Tests the equivalence of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

References [std::sub_match<_BiIter>::compare\(\)](#).

operator==() [7/8]

```
template<typename _Bi_iter >
bool operator== (
    typename iterator_traits< _Bi_iter >::value_type const & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the equivalence of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

References [std::__addressof\(\)](#).

operator==() [8/8]

```
template<typename _Bi_iter >
bool operator== (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the equivalence of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

References [std::sub_match<_Biter>::compare\(\)](#).

operator>() [1/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator> (
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub\_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>() [2/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator> (
    const sub\_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>() [3/7]

```
template<typename _Bi_iter >
bool operator> (
    const sub\_match< _Bi_iter > & __lhs,
    typename iterator\_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>() [4/7]

```
template<typename _Bi_iter >
bool operator> (
    const sub\_match< _Bi_iter > & __lhs,
    typename iterator\_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a C string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>() [5/7]

```
template<typename _BiIter >
bool operator> (
    const sub\_match< _BiIter > & __lhs,
    const sub\_match< _BiIter > & __rhs ) [related]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

References [std::sub_match<_BiIter>::compare\(\)](#).

operator>() [6/7]

```
template<typename _Bi_iter >
bool operator> (
    typename iterator\_traits< _Bi_iter >::value_type const & __lhs,
    const sub\_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>() [7/7]

```
template<typename _Bi_iter >
bool operator> (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

operator>=() [1/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator>= (
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

operator>=() [2/7]

```
template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >
bool operator>= (
    const sub_match< _Bi_iter > & __lhs,
    const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

operator>=() [3/7]

```
template<typename _Bi_iter >
bool operator>= (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const & __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a character.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A character.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

operator>=() [4/7]

```
template<typename _Bi_iter >
bool operator>= (
    const sub_match< _Bi_iter > & __lhs,
    typename iterator_traits< _Bi_iter >::value_type const * __rhs ) [related]
```

Tests the ordering of a regular expression submatch and a C string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A null-terminated string.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

operator>=() [5/7]

```
template<typename _BiIter >
bool operator>= (
    const sub_match< _BiIter > & __lhs,
    const sub_match< _BiIter > & __rhs ) [related]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

References [std::sub_match<_Biter>::compare\(\)](#).

operator>=() [6/7]

```
template<typename _Bi_iter >
bool operator>= (
    typename iterator_traits< _Bi_iter >::value_type const & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a character and a regular expression submatch.

Parameters

<code>__lhs</code>	A character.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

operator>=() [7/7]

```
template<typename _Bi_iter >
bool operator>= (
    typename iterator_traits< _Bi_iter >::value_type const * __lhs,
    const sub_match< _Bi_iter > & __rhs ) [related]
```

Tests the ordering of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A null-terminated string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

regex_match() [1/7]

```
template<typename _Bi_iter , typename _Ch_type , typename _Rx_traits >
bool std::regex_match (
    _Bi_iter __first,
    _Bi_iter __last,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Indicates if there is a match between the regular expression `e` and all of the character sequence `[first, last)`.

Parameters

<code>__first</code>	Beginning of the character sequence to match.
----------------------	---

Parameters

<code>__last</code>	One-past-the-end of the character sequence to match.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

References [std::regex_match\(\)](#).

regex_match() [2/7]

```
template<typename _Bi_iter , typename _Alloc , typename _Ch_type , typename _Rx_traits >
bool std::regex_match (
    _Bi_iter __s,
    _Bi_iter __e,
    match_results< _Bi_iter, _Alloc > & __m,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Determines if there is a match between the regular expression `e` and all of the character sequence `[first, last)`.

Parameters

<code>__s</code>	Start of the character sequence to match.
<code>__e</code>	One-past-the-end of the character sequence to match.
<code>__m</code>	The match results.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Referenced by [std::regex_match\(\)](#), [std::regex_match\(\)](#), [std::regex_match\(\)](#), [std::regex_match\(\)](#), and [std::regex_match\(\)](#).

regex_match() [3/7]

```
template<typename _Ch_type , class _Rx_traits >
bool std::regex_match (
    const _Ch_type * __s,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Indicates if there is a match between the regular expression *e* and a C-style null-terminated string.

Parameters

<code>__s</code>	The C-style null-terminated string to match.
<code>__re</code>	The regular expression.
<code>__f</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

References [std::regex_match\(\)](#).

regex_match() [4/7]

```
template<typename _Ch_type , typename _Alloc , typename _Rx_traits >
bool std::regex_match (
    const _Ch_type * __s,
    match_results< const _Ch_type *, _Alloc > & __m,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Determines if there is a match between the regular expression *e* and a C-style null-terminated string.

Parameters

<code>__s</code>	The C-style null-terminated string to match.
<code>__m</code>	The match results.
<code>__re</code>	The regular expression.
<code>__f</code>	Controls how the regular expression is matched.

Return values

<i>true</i>	A match exists.
<i>false</i>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::regex_match\(\)](#).

regex_match() [5/7]

```
template<typename _Ch_traits , typename _Ch_alloc , typename _Alloc , typename _Ch_type , typename
_Rx_traits >
bool std::regex_match (
    const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > && ,
    match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator,
    _Alloc > & ,
    const basic_regex< _Ch_type, _Rx_traits > & ,
    regex_constants::match_flag_type = regex_constants::match_default ) [delete]
```

Prevent unsafe attempts to get `match_results` from a temporary string.

regex_match() [6/7]

```
template<typename _Ch_traits , typename _Ch_alloc , typename _Alloc , typename _Ch_type , typename
_Rx_traits >
bool std::regex_match (
    const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s,
    match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator,
    _Alloc > & __m,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Determines if there is a match between the regular expression `e` and a string.

Parameters

<code>__s</code>	The string to match.
<code>__m</code>	The match results.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<i>true</i>	A match exists.
<i>false</i>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::basic_string<_CharT, _Traits, _Alloc>::begin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::end\(\)](#), and [std::regex_match\(\)](#).

regex_match() [7/7]

```
template<typename _Ch_traits , typename _Str_allocator , typename _Ch_type , typename _Rx_traits
>
bool std::regex_match (
    const basic_string< _Ch_type, _Ch_traits, _Str_allocator > & __s,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Indicates if there is a match between the regular expression *e* and a string.

Parameters

<i>__s</i>	[IN] The string to match.
<i>__re</i>	[IN] The regular expression.
<i>__flags</i>	[IN] Controls how the regular expression is matched.

Return values

<i>true</i>	A match exists.
<i>false</i>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::basic_string<_CharT, _Traits, _Alloc>::begin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::end\(\)](#), and [std::regex_match\(\)](#).

regex_replace() [1/6]

```
template<typename _Out_iter , typename _Bi_iter , typename _Rx_traits , typename _Ch_type >
_Out_iter std::regex_replace (
    _Out_iter __out,
    _Bi_iter __first,
    _Bi_iter __last,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const _Ch_type * __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default )
```

Search for a regular expression within a range for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<i>__out</i>	[OUT] The output iterator.
<i>__first</i>	[IN] The start of the string to search.
<i>__last</i>	[IN] One-past-the-end of the string to search.
<i>__e</i>	[IN] The regular expression to search for.
<i>__fmt</i>	[IN] The format C-string.

Parameters

<code>__flags</code>	[IN] Search and replace policy flags.
----------------------	---------------------------------------

Returns

`__out`

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::__regex_replace\(\)](#).

regex_replace() [2/6]

```
template<typename _Out_iter , typename _Bi_iter , typename _Rx_traits , typename _Ch_type , typename
_St , typename _Sa >
_Out_iter std::regex_replace (
    _Out_iter __out,
    _Bi_iter __first,
    _Bi_iter __last,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const basic_string< _Ch_type, _St, _Sa > & __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a range for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__out</code>	[OUT] The output iterator.
<code>__first</code>	[IN] The start of the string to search.
<code>__last</code>	[IN] One-past-the-end of the string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

`__out`

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::__regex_replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>](#)
Referenced by [std::regex_replace\(\)](#), [std::regex_replace\(\)](#), [std::regex_replace\(\)](#), and [std::regex_replace\(\)](#).

regex_replace() [3/6]

```
template<typename _Rx_traits , typename _Ch_type >
basic_string< _Ch_type > std::regex_replace (
    const _Ch_type * __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const _Ch_type * __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a C-string for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<code>__s</code>	[IN] The C-string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format C-string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::back_inserter\(\)](#), and [std::regex_replace\(\)](#).

regex_replace() [4/6]

```
template<typename _Rx_traits , typename _Ch_type , typename _St , typename _Sa >
basic_string< _Ch_type > std::regex_replace (
    const _Ch_type * __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const basic_string< _Ch_type, _St, _Sa > & __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a C-string for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__s</code>	[IN] The C-string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::back_inserter\(\)](#), and [std::regex_replace\(\)](#).

regex_replace() [5/6]

```
template<typename _Rx_traits , typename _Ch_type , typename _St , typename _Sa >
basic_string< _Ch_type, _St, _Sa > std::regex_replace (
    const basic_string< _Ch_type, _St, _Sa > & __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const _Ch_type * __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a string for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<code>__s</code>	[IN] The string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format C-string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::back_inserter\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::begin\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::end\(\)](#) and [std::regex_replace\(\)](#).

regex_replace() [6/6]

```
template<typename _Rx_traits , typename _Ch_type , typename _St , typename _Sa , typename _Fst ,
typename _Fsa >
basic_string< _Ch_type, _St, _Sa > std::regex_replace (
    const basic_string< _Ch_type, _St, _Sa > & __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    const basic_string< _Ch_type, _Fst, _Fsa > & __fmt,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a string for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__s</code>	[IN] The string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.

Parameters

<code>__flags</code>	[IN] Search and replace policy flags.
----------------------	---------------------------------------

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::back_inserter\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::begin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::end\(\)](#) and [std::regex_replace\(\)](#).

regex_search() [1/7]

```
template<typename _Bi_iter , typename _Ch_type , typename _Rx_traits >
bool std::regex_search (
    _Bi_iter __first,
    _Bi_iter __last,
    const basic_regex< _Ch_type, _Rx_traits > & __re,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a range.

Parameters

<code>__first</code>	[IN] The start of the string to search.
<code>__last</code>	[IN] One-past-the-end of the string to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<i>true</i>	A match was found within the string.
<i>false</i>	No match was found within the string.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::regex_search\(\)](#).

regex_search() [2/7]

```
template<typename _Bi_iter , typename _Alloc , typename _Ch_type , typename _Rx_traits >
bool std::regex_search (
    _Bi_iter __s,
    _Bi_iter __e,
```

```
match_results< _Bi_iter, _Alloc > & __m,
const basic_regex< _Ch_type, _Rx_traits > & __re,
regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a range.

Parameters

<code>__s</code>	[IN] The start of the string to search.
<code>__e</code>	[IN] One-past-the-end of the string to search.
<code>__m</code>	[OUT] The match results.
<code>__re</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of <code>m</code> is undefined.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Referenced by `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_iterator()`, `std::regex_search()`, `std::regex_search()`, `std::regex_search()`, `std::regex_search()`, and `std::regex_search()`.

regex_search() [3/7]

```
template<typename _Ch_type , typename _Rx_traits >
bool std::regex_search (
    const _Ch_type * __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a C-string.

Parameters

<code>__s</code>	[IN] The C-string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__f</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::regex_search\(\)](#).

regex_search() [4/7]

```
template<typename _Ch_type , class _Alloc , class _Rx_traits >
bool std::regex_search (
    const _Ch_type * __s,
    match_results< const _Ch_type *, _Alloc > & __m,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a C-string.

Parameters

<code>__s</code>	[IN] A C-string to search for the regex.
<code>__m</code>	[OUT] The set of regex matches.
<code>__e</code>	[IN] The regex to search for in <code>s</code> .
<code>__f</code>	[IN] The search flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of <code>m</code> is undefined.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::regex_search\(\)](#).

regex_search() [5/7]

```
template<typename _Ch_traits , typename _Ch_alloc , typename _Alloc , typename _Ch_type , typename
_Rx_traits >
bool std::regex_search (
    const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > && ,
    match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > & ,
    const basic_regex< _Ch_type, _Rx_traits > & ,
    regex_constants::match_flag_type = regex_constants::match_default ) [delete]
```

Prevent unsafe attempts to get `match_results` from a temporary string.

regex_search() [6/7]

```
template<typename _Ch_traits , typename _Ch_alloc , typename _Alloc , typename _Ch_type , typename
_Rx_traits >
bool std::regex_search (
    const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s,
    match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > & __m,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a string.

Parameters

<code>__s</code>	[IN] A C++ string to search for the regex.
<code>__m</code>	[OUT] The set of regex matches.
<code>__e</code>	[IN] The regex to search for in s.
<code>__f</code>	[IN] The search flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of m is undefined.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

References `std::basic_string<_CharT, _Traits, _Alloc>::begin()`, `std::basic_string<_CharT, _Traits, _Alloc>::end()`, and `std::regex_search()`.

regex_search() [7/7]

```
template<typename _Ch_traits , typename _String_allocator , typename _Ch_type , typename _Rx_traits >
bool std::regex_search (
    const basic_string< _Ch_type, _Ch_traits, _String_allocator > & __s,
    const basic_regex< _Ch_type, _Rx_traits > & __e,
    regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a string.

Parameters

<code>__s</code>	[IN] The string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<i>true</i>	A match was found within the string.
<i>false</i>	No match was found within the string.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

References [std::regex_search\(\)](#).

swap() [1/2]

```
template<typename _Ch_type , typename _Rx_traits >
void swap (
    basic_regex< _Ch_type, _Rx_traits > & __lhs,
    basic_regex< _Ch_type, _Rx_traits > & __rhs ) [related]
```

Swaps the contents of two regular expression objects.

Parameters

<i>__lhs</i>	First regular expression.
<i>__rhs</i>	Second regular expression.

swap() [2/2]

```
template<typename _Bi_iter , typename _Alloc >
void std::swap (
    match_results< _Bi_iter, _Alloc > & __lhs,
    match_results< _Bi_iter, _Alloc > & __rhs ) [inline], [noexcept]
```

Swaps two match results.

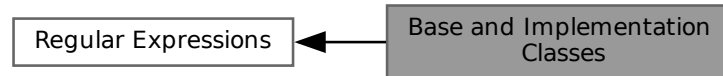
Parameters

<i>__lhs</i>	A match result.
<i>__rhs</i>	A match result.

The contents of the two `match_results` objects are swapped.

3.13.4 Base and Implementation Classes

Collaboration diagram for Base and Implementation Classes:



Classes

- struct `std::__detail::__BracketMatcher< _TraitsT, __icase, __collate >`
- class `std::__detail::__Compiler< _TraitsT >`
- class `std::__detail::__Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode >`
- class `std::__detail::__Scanner< _CharT >`
- class `std::__detail::__StateSeq< _TraitsT >`

Typedefs

- template<typename `_CharT` >
using `std::__detail::__Matcher` = `std::function< bool(_CharT)>`
- typedef long `std::__detail::__StateldT`

Enumerations

- enum `std::__detail::__Opcode` : int {
`_S_opcode_unknown` , `_S_opcode_alternative` , `_S_opcode_repeat` , `_S_opcode_backref` ,
`_S_opcode_line_begin_assertion` , `_S_opcode_line_end_assertion` , `_S_opcode_word_boundary` , `_S_opcode_subexpr_lookahead` ,
`_S_opcode_subexpr_begin` , `_S_opcode_subexpr_end` , `_S_opcode_dummy` , `_S_opcode_match` ,
`_S_opcode_accept` }

Variables

- constexpr `_StateldT std::__detail::__S_invalid_state_id`

3.13.4.1 Detailed Description

3.13.4.2 Enumeration Type Documentation

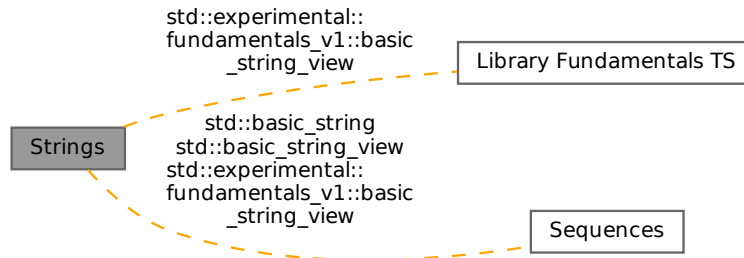
`_Opcode`

```
enum std::__detail::__Opcode : int
```

Operation codes that define the type of transitions within the base NFA that represents the regular expression.

3.14 Strings

Collaboration diagram for Strings:



Classes

- class `std::basic_string<_CharT, _Traits, _Alloc>`
- class `std::basic_string_view<_CharT, _Traits>`
- class `std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits>`
- struct `std::char_traits<_CharT>`

Typedefs

- typedef `basic_string<char>` `std::string`
- typedef `basic_string<char16_t>` `std::u16string`
- typedef `basic_string<char32_t>` `std::u32string`
- typedef `basic_string<wchar_t>` `std::wstring`

3.14.1 Detailed Description

3.14.2 Typedef Documentation

string

```
typedef basic_string<char> std::string
```

A string of `char`.

u16string

```
typedef basic_string<char16_t> std::u16string
```

A string of `char16_t`.

u32string

```
typedef basic_string<char32_t> std::u32string
```

A string of `char32_t`.

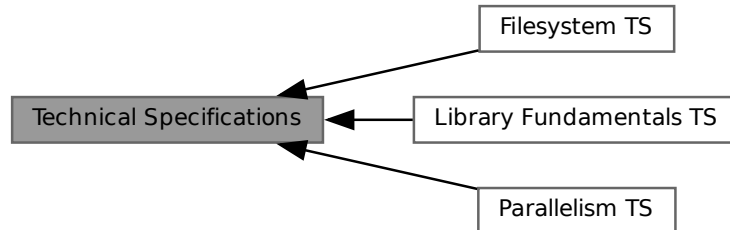
wstring

```
typedef basic_string<wchar_t> std::wstring
```

A string of `wchar_t`.

3.15 Technical Specifications

Collaboration diagram for Technical Specifications:



Modules

- [Filesystem TS](#)
- [Library Fundamentals TS](#)
- [Parallelism TS](#)

3.15.1 Detailed Description

Components specified by various Technical Specifications.

As indicated by the `std::experimental` namespace and the header paths, the contents of these Technical Specifications are experimental and not part of the C++ standard. As such the interfaces and implementations may change in the future, and there is **no guarantee of compatibility between different GCC releases** for these features.

3.15.2 Filesystem TS

Collaboration diagram for Filesystem TS:



Files

- file [experimental/filesystem](#)

Classes

- class [std::experimental::filesystem::v1::filesystem_error](#)
- class [std::experimental::filesystem::v1::path::iterator](#)

- class `std::experimental::filesystem::v1::path`
- struct `std::experimental::filesystem::v1::space_info`

Typedefs

- using `std::experimental::filesystem::file_time_type` = `std::chrono::system_clock::time_point`

Enumerations

- enum class `std::experimental::filesystem::copy_options` : unsigned short { `none`, `skip_existing`, `overwrite_existing`, `update_existing`, `recursive`, `copy_symlinks`, `skip_symlinks`, `directories_only`, `create_symlinks`, `create_hard_links` }
- enum class `std::experimental::filesystem::directory_options` : unsigned char { `none`, `follow_directory_symlink`, `skip_permission_denied` }
- enum class `std::experimental::filesystem::file_type` : signed char { `none`, `not_found`, `regular`, `directory`, `symlink`, `block`, `character`, `fifo`, `socket`, `unknown` }
- enum class `std::experimental::filesystem::perms` : unsigned { `none`, `owner_read`, `owner_write`, `owner_exec`, `owner_all`, `group_read`, `group_write`, `group_exec`, `group_all`, `others_read`, `others_write`, `others_exec`, `others_all`, `all`, `set_uid`, `set_gid`, `sticky_bit`, `mask`, `unknown`, `add_perms`, `remove_perms`, `symlink_nofollow` }

Functions

- `std::experimental::filesystem::v1::path::path` (const `path` &__p)
- `std::experimental::filesystem::v1::path::path` (`path` &&__p) noexcept
- `std::experimental::filesystem::v1::path::path` (`string_type` &&__source)
- `path` `std::experimental::filesystem::absolute` (const `path` &__p, const `path` &__base=current_path())
- `path` & `std::experimental::filesystem::v1::path::assign` (`string_type` &&__source)
- `iterator` `std::experimental::filesystem::v1::path::begin` () const noexcept
- `directory_iterator` `std::experimental::filesystem::begin` (`directory_iterator` __iter) noexcept
- `recursive_directory_iterator` `std::experimental::filesystem::begin` (`recursive_directory_iterator` __iter) noexcept
- `path` `std::experimental::filesystem::canonical` (const `path` &__p, const `path` &__base, `error_code` &__ec)
- `path` `std::experimental::filesystem::canonical` (const `path` &__p, const `path` &__base=current_path())
- `path` `std::experimental::filesystem::canonical` (const `path` &__p, `error_code` &__ec)
- int `std::experimental::filesystem::v1::path::compare` (const `basic_string_view`< value_type > __s) const
- int `std::experimental::filesystem::v1::path::compare` (const `string_type` &__s) const
- int `std::experimental::filesystem::v1::path::compare` (const value_type *__s) const
- void `std::experimental::filesystem::copy` (const `path` &__from, const `path` &__to)
- void `std::experimental::filesystem::copy` (const `path` &__from, const `path` &__to, `copy_options` __options)
- void `std::experimental::filesystem::copy` (const `path` &__from, const `path` &__to, `copy_options` __options, `error_code` &) noexcept
- void `std::experimental::filesystem::copy` (const `path` &__from, const `path` &__to, `error_code` &__ec) noexcept
- bool `std::experimental::filesystem::copy_file` (const `path` &__from, const `path` &__to)
- bool `std::experimental::filesystem::copy_file` (const `path` &__from, const `path` &__to, `copy_options` __option)
- bool `std::experimental::filesystem::copy_file` (const `path` &__from, const `path` &__to, `copy_options` __option, `error_code` &)
- bool `std::experimental::filesystem::copy_file` (const `path` &__from, const `path` &__to, `error_code` &__ec)

- void **std::experimental::filesystem::copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_↵
symlink)
- void **std::experimental::filesystem::copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_↵
symlink, [error_code](#) &__ec) noexcept
- bool **std::experimental::filesystem::create_directories** (const [path](#) &__p)
- bool **std::experimental::filesystem::create_directories** (const [path](#) &__p, [error_code](#) &__ec)
- bool **std::experimental::filesystem::create_directory** (const [path](#) &__p)
- bool **std::experimental::filesystem::create_directory** (const [path](#) &__p, const [path](#) &attributes)
- bool **std::experimental::filesystem::create_directory** (const [path](#) &__p, const [path](#) &attributes, [error_code](#) &↵
__ec) noexcept
- bool **std::experimental::filesystem::create_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- void **std::experimental::filesystem::create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **std::experimental::filesystem::create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink,
[error_code](#) &__ec) noexcept
- void **std::experimental::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link)
- void **std::experimental::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link,
[error_code](#) &__ec) noexcept
- void **std::experimental::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **std::experimental::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink,
[error_code](#) &__ec) noexcept
- [path](#) **std::experimental::filesystem::current_path** ()
- void **std::experimental::filesystem::current_path** (const [path](#) &__p)
- void **std::experimental::filesystem::current_path** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) **std::experimental::filesystem::current_path** ([error_code](#) &__ec)
- [iterator](#) **std::experimental::filesystem::v1::path::end** () const noexcept
- [directory_iterator](#) **std::experimental::filesystem::end** ([directory_iterator](#)) noexcept
- [recursive_directory_iterator](#) **std::experimental::filesystem::end** ([recursive_directory_iterator](#)) noexcept
- bool **std::experimental::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2)
- bool **std::experimental::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2, [error_code](#) &__ec) noex-
cept
- bool **std::experimental::filesystem::exists** (const [path](#) &__p)
- bool **std::experimental::filesystem::exists** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::experimental::filesystem::exists** ([file_status](#) __s) noexcept
- [path](#) **std::experimental::filesystem::v1::path::extension** () const
- [uintmax_t](#) **std::experimental::filesystem::file_size** (const [path](#) &__p)
- [uintmax_t](#) **std::experimental::filesystem::file_size** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) **std::experimental::filesystem::v1::path::filename** () const
- [std::string](#) **std::experimental::filesystem::v1::path::generic_string** () const
- [template](#)<[typename](#) _CharT, [typename](#) _Traits = [std::char_traits](#)<_CharT>, [typename](#) _Allocator = [std::allocator](#)<_CharT>>
[std::basic_string](#)<_CharT, _Traits, _Allocator> **std::experimental::filesystem::v1::path::generic_string**
(const _Allocator &__a= _Allocator()) const
- [std::u16string](#) **std::experimental::filesystem::v1::path::generic_u16string** () const
- [std::u32string](#) **std::experimental::filesystem::v1::path::generic_u32string** () const
- [std::string](#) **std::experimental::filesystem::v1::path::generic_u8string** () const
- [std::wstring](#) **std::experimental::filesystem::v1::path::generic_wstring** () const
- [uintmax_t](#) **std::experimental::filesystem::hard_link_count** (const [path](#) &__p)
- [uintmax_t](#) **std::experimental::filesystem::hard_link_count** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::experimental::filesystem::v1::path::has_extension** () const
- bool **std::experimental::filesystem::v1::path::has_stem** () const
- bool **std::experimental::filesystem::v1::path::is_absolute** () const
- bool **std::experimental::filesystem::is_block_file** (const [path](#) &__p)

- `bool std::experimental::filesystem::is_block_file (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_block_file (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_character_file (const path &__p)`
- `bool std::experimental::filesystem::is_character_file (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_character_file (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_directory (const path &__p)`
- `bool std::experimental::filesystem::is_directory (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_directory (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_empty (const path &__p)`
- `bool std::experimental::filesystem::is_empty (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_fifo (const path &__p)`
- `bool std::experimental::filesystem::is_fifo (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_fifo (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_other (const path &__p)`
- `bool std::experimental::filesystem::is_other (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_other (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_regular_file (const path &__p)`
- `bool std::experimental::filesystem::is_regular_file (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_regular_file (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_socket (const path &__p)`
- `bool std::experimental::filesystem::is_socket (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_socket (file_status __s) noexcept`
- `bool std::experimental::filesystem::is_symlink (const path &__p)`
- `bool std::experimental::filesystem::is_symlink (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::is_symlink (file_status __s) noexcept`
- `file_time_type std::experimental::filesystem::last_write_time (const path &__p)`
- `file_time_type std::experimental::filesystem::last_write_time (const path &__p, error_code &__ec) noexcept`
- `void std::experimental::filesystem::last_write_time (const path &__p, file_time_type __new_time)`
- `void std::experimental::filesystem::last_write_time (const path &__p, file_time_type __new_time, error_code &__ec) noexcept`
- `path & std::experimental::filesystem::v1::path::make_preferred ()`
- `bool std::experimental::filesystem::operator!= (const directory_iterator &__lhs, const directory_iterator &__rhs)`
- `bool std::experimental::filesystem::operator!= (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)`
- `copy_options & std::experimental::filesystem::operator&= (copy_options &__x, copy_options __y) noexcept`
- `reference std::experimental::filesystem::v1::path::iterator::operator* () const noexcept`
- `iterator & std::experimental::filesystem::v1::path::iterator::operator++ () noexcept`
- `template<typename _CharT>
__detail::_Path<_CharT*, _CharT*> & std::experimental::filesystem::v1::path::operator+= (_CharT __x)`
- `path & std::experimental::filesystem::v1::path::operator+= (basic_string_view<value_type> __x)`
- `path & std::experimental::filesystem::v1::path::operator+= (const path &__x)`
- `path & std::experimental::filesystem::v1::path::operator+= (const string_type &__x)`
- `path & std::experimental::filesystem::v1::path::operator+= (const value_type *__x)`
- `path & std::experimental::filesystem::v1::path::operator+= (value_type __x)`
- `iterator & std::experimental::filesystem::v1::path::iterator::operator-- () noexcept`
- `bool std::experimental::filesystem::operator< (const path &__lhs, const path &__rhs) noexcept`
- `path & std::experimental::filesystem::v1::path::operator= (const path &__p)`
- `path & std::experimental::filesystem::v1::path::operator= (path &&__p) noexcept`
- `path & std::experimental::filesystem::v1::path::operator= (string_type &&__source)`

- `bool std::experimental::filesystem::operator== (const directory_iterator &__lhs, const directory_iterator &__rhs)`
- `bool std::experimental::filesystem::operator== (const path &__lhs, const path &__rhs) noexcept`
- `bool std::experimental::filesystem::operator== (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)`
- `constexpr copy_options std::experimental::filesystem::operator^ (copy_options __x, copy_options __y) noexcept`
- `copy_options & std::experimental::filesystem::operator^= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options std::experimental::filesystem::operator| (copy_options __x, copy_options __y) noexcept`
- `copy_options & std::experimental::filesystem::operator|= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options std::experimental::filesystem::operator~ (copy_options __x) noexcept`
- `void std::experimental::filesystem::permissions (const path &__p, perms __prms)`
- `void std::experimental::filesystem::permissions (const path &__p, perms __prms, error_code &__ec) noexcept`
- `path std::experimental::filesystem::read_symlink (const path &__p)`
- `path std::experimental::filesystem::read_symlink (const path &__p, error_code &__ec)`
- `bool std::experimental::filesystem::remove (const path &__p)`
- `bool std::experimental::filesystem::remove (const path &__p, error_code &__ec) noexcept`
- `uintmax_t std::experimental::filesystem::remove_all (const path &__p)`
- `uintmax_t std::experimental::filesystem::remove_all (const path &__p, error_code &__ec)`
- `void std::experimental::filesystem::rename (const path &__from, const path &__to)`
- `void std::experimental::filesystem::rename (const path &__from, const path &__to, error_code &__ec) noexcept`
- `void std::experimental::filesystem::resize_file (const path &__p, uintmax_t __size)`
- `void std::experimental::filesystem::resize_file (const path &__p, uintmax_t __size, error_code &__ec) noexcept`
- `space_info std::experimental::filesystem::space (const path &__p)`
- `space_info std::experimental::filesystem::space (const path &__p, error_code &__ec) noexcept`
- `file_status std::experimental::filesystem::status (const path &__p)`
- `file_status std::experimental::filesystem::status (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::status_known (file_status) noexcept`
- `path std::experimental::filesystem::v1::path::stem () const`
- `std::string std::experimental::filesystem::v1::path::string () const`
- `template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
std::basic_string<_CharT, _Traits, _Allocator> std::experimental::filesystem::v1::path::string (const _CharT* __a= _Allocator()) const`
- `void std::experimental::filesystem::v1::path::swap (path &__rhs) noexcept`
- `file_status std::experimental::filesystem::symlink_status (const path &__p)`
- `file_status std::experimental::filesystem::symlink_status (const path &__p, error_code &__ec) noexcept`
- `path std::experimental::filesystem::system_complete (const path &__p)`
- `path std::experimental::filesystem::system_complete (const path &__p, error_code &__ec)`
- `path std::experimental::filesystem::temp_directory_path ()`
- `path std::experimental::filesystem::temp_directory_path (error_code &__ec)`
- `std::u16string std::experimental::filesystem::v1::path::u16string () const`
- `std::u32string std::experimental::filesystem::v1::path::u32string () const`
- `std::string std::experimental::filesystem::v1::path::u8string () const`
- `std::wstring std::experimental::filesystem::v1::path::wstring () const`
- `constexpr perms std::experimental::filesystem::operator| (perms __x, perms __y) noexcept`
- `constexpr perms std::experimental::filesystem::operator^ (perms __x, perms __y) noexcept`
- `constexpr perms std::experimental::filesystem::operator~ (perms __x) noexcept`

- `perms & std::experimental::filesystem::operator&= (perms &__x, perms __y) noexcept`
- `perms & std::experimental::filesystem::operator|= (perms &__x, perms __y) noexcept`
- `perms & std::experimental::filesystem::operator^= (perms &__x, perms __y) noexcept`
- `constexpr directory_options std::experimental::filesystem::operator| (directory_options __x, directory_options __y) noexcept`
- `constexpr directory_options std::experimental::filesystem::operator^ (directory_options __x, directory_options __y) noexcept`
- `constexpr directory_options std::experimental::filesystem::operator~ (directory_options __x) noexcept`
- `directory_options & std::experimental::filesystem::operator&= (directory_options &__x, directory_options __y) noexcept`
- `directory_options & std::experimental::filesystem::operator|= (directory_options &__x, directory_options __y) noexcept`
- `directory_options & std::experimental::filesystem::operator^= (directory_options &__x, directory_options __y) noexcept`

3.15.2.1 Detailed Description

Utilities for performing operations on file systems and their components, such as paths, regular files, and directories. ISO/IEC TS 18822:2015 C++ File System Technical Specification

Since

C++11

Remarks

Link using `-lstdc++fs` to use these types and functions.

3.15.2.2 Typedef Documentation

`file_time_type`

```
using std::experimental::filesystem::v1::file_time_type = typedef std::chrono::system_clock<
::time_point
```

The type used for file timestamps.

3.15.2.3 Enumeration Type Documentation

`copy_options`

```
enum class std::experimental::filesystem::v1::copy_options : unsigned short [strong]
```

Bitmask type controlling effects of `filesystem::copy`

`directory_options`

```
enum class std::experimental::filesystem::v1::directory_options : unsigned char [strong]
```

Bitmask type controlling directory iteration.

`file_type`

```
enum class std::experimental::filesystem::v1::file_type : signed char [strong]
```

Enumerated type representing the type of a file.

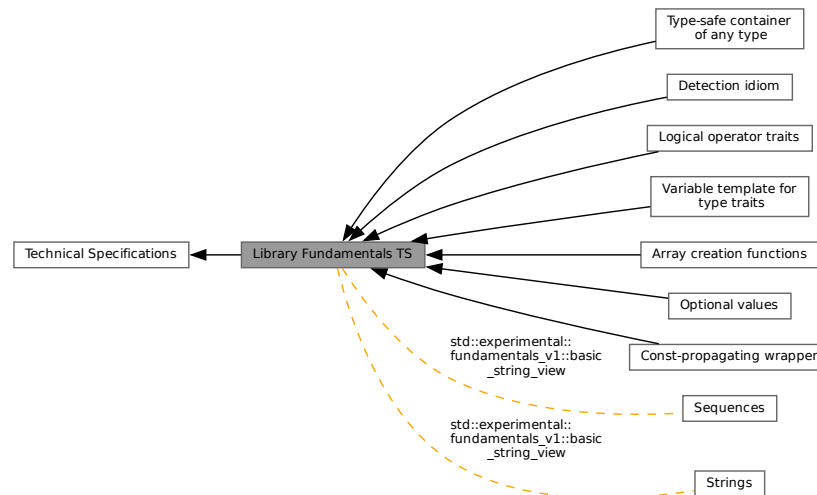
`perms`

```
enum class std::experimental::filesystem::v1::perms : unsigned [strong]
```

Bitmask type representing file access permissions.

3.15.3 Library Fundamentals TS

Collaboration diagram for Library Fundamentals TS:



Modules

- [Array creation functions](#)
- [Const-propagating wrapper](#)
- [Detection idiom](#)
- [Logical operator traits](#)
- [Optional values](#)
- [Type-safe container of any type](#)
- [Variable template for type traits](#)

Files

- file [experimental/algorithm](#)
- file [experimental/any](#)
- file [experimental/array](#)
- file [experimental/chrono](#)
- file [experimental/deque](#)
- file [experimental/forward_list](#)
- file [experimental/functional](#)
- file [experimental/iterator](#)
- file [experimental/list](#)
- file [experimental/map](#)
- file [experimental/memory](#)
- file [experimental/memory_resource](#)
- file [experimental/numeric](#)
- file [experimental/optional](#)
- file [propagate_const](#)
- file [experimental/random](#)

- file [experimental/ratio](#)
- file [experimental/regex](#)
- file [experimental/set](#)
- file [experimental/string](#)
- file [experimental/string_view](#)
- file [experimental/system_error](#)
- file [experimental/tuple](#)
- file [experimental/type_traits](#)
- file [experimental/unordered_map](#)
- file [experimental/unordered_set](#)
- file [experimental/utility](#)
- file [experimental/vector](#)

Classes

- class [std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits>](#)

3.15.3.1 Detailed Description

Components defined by the *C++ Extensions for Library Fundamentals* Technical Specification, versions 1 and 2.

- ISO/IEC TS 19568:2015 C++ Extensions for Library Fundamentals
- ISO/IEC TS 19568:2017 C++ Extensions for Library Fundamentals, Version 2

3.15.3.2 Array creation functions

Collaboration diagram for Array creation functions:



Functions

- `template<typename _Tp, size_t _Nm, size_t... _Idx>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::__to_array (_Tp(&__a)[_Nm],
index_sequence< _Idx... >)`
- `template<typename _Dest = void, typename... _Types>
constexpr array< typename __make_array_elem< _Dest, _Types... >::type, sizeof...(_Types)> std::experimental::make_array
(_Types &&... __t)`
- `template<typename _Tp, size_t _Nm>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::to_array (_Tp(&__a)[_Nm]) noexcept(is_nothrow_constructible<
remove_cv_t< _Tp >, _Tp & >::value)`

3.15.3.2.1 Detailed Description

Array creation functions as described in N4529, Working Draft, C++ Extensions for Library Fundamentals, Version 2

3.15.3.2.2 Function Documentation

make_array()

```
template<typename _Dest = void, typename... _Types>
constexpr array< typename __make_array_elem< _Dest, _Types... >::type, sizeof...(_Types)> std::
::experimental::fundamentals_v2::make_array (
    _Types &&... __t ) [constexpr]
```

Create a `std::array` from a variable-length list of arguments.

to_array()

```
template<typename _Tp , size_t _Nm>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::fundamentals_v2::to_array (
    _Tp(&) __a[_Nm] ) [constexpr], [noexcept]
```

Create a `std::array` from an array.

3.15.3.3 Const-propagating wrapper

Collaboration diagram for Const-propagating wrapper:



Classes

- class `std::experimental::fundamentals_v2::propagate_const< _Tp >`

Functions

- `template<typename _Tp >`
`constexpr const _Tp & std::experimental::get_underlying (const propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Tp >`
`constexpr _Tp & std::experimental::get_underlying (propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Tp , typename _Up >`
`constexpr bool std::experimental::operator!= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp , typename _Up >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp , typename _Up >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator!= (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp , typename _Up >`
`constexpr bool std::experimental::operator< (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp , typename _Up >`
`constexpr bool std::experimental::operator< (const propagate_const< _Tp > &__pt, const _Up &__u)`

- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator< (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator== (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr enable_if_t< __is_swappable< _Tp >::value, void > std::experimental::swap (propagate_const< _Tp > &__pt, propagate_const< _Tp > &__pt2) noexcept(__is_nothrow_swappable< _Tp >::value)`

3.15.3.3.1 Detailed Description

A const-propagating wrapper that propagates const to pointer-like members, as described in n4388 “A Proposal to Add a Const-Propagating Wrapper to the Standard Library”.

3.15.3.4 Detection idiom

Collaboration diagram for Detection idiom:



- `#define __cpp_lib_experimental_detect`
- `template<typename... >`
 `using std::experimental::void_t = void`
- `template<template< typename... > class _Op, typename... _Args>`
 `using std::experimental::is_detected = typename std::__detector< nonesuch, void, _Op, _Args... >::value_t`
- `template<template< typename... > class _Op, typename... _Args>`
 `using std::experimental::detected_t = typename std::__detector< nonesuch, void, _Op, _Args... >::type`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
 `using std::experimental::detected_or = std::__detected_or< _Default, _Op, _Args... >`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
 `using std::experimental::detected_or_t = typename detected_or< _Default, _Op, _Args... >::type`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
 `using std::experimental::is_detected_exact = is_same< _Expected, detected_t< _Op, _Args... > >`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
 `using std::experimental::is_detected_convertible = is_convertible< detected_t< _Op, _Args... >, _To >`
- `template<template< typename... > class _Op, typename... _Args>`
 `constexpr bool std::experimental::is_detected_v`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
 `constexpr bool std::experimental::is_detected_exact_v`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
 `constexpr bool std::experimental::is_detected_convertible_v`

3.15.3.4.1 Detailed Description

Since

Library Fundamentals TS v2. C++14.

3.15.3.4.2 Macro Definition Documentation

`__cpp_lib_experimental_detect`

```
#define __cpp_lib_experimental_detect
```

A metafunction that always yields void, used for detecting valid types.

3.15.3.4.3 Typedef Documentation

`detected_or`

```
template<typename _Default, template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals_v2::detected_or = typedef std::__detected_or<_Default, _Op,
_Args...>
```

A metafunction that always yields void, used for detecting valid types.

detected_or_t

```
template<typename _Default , template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals\_v2::detected\_or\_t = typedef typename detected_or<_Default,
_Op, _Args...>::type
```

A metafunction that always yields void, used for detecting valid types.

detected_t

```
template<template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals\_v2::detected\_t = typedef typename std::__detector<nonesuch,
void, _Op, _Args...>::type
```

A metafunction that always yields void, used for detecting valid types.

is_detected

```
template<template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals\_v2::is\_detected = typedef typename std::__detector<nonesuch,
void, _Op, _Args...>::value_t
```

A metafunction that always yields void, used for detecting valid types.

is_detected_convertible

```
template<typename _To , template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals\_v2::is\_detected\_convertible = typedef is\_convertible<detected_t<
_t<_Op, _Args...>, _To>
```

A metafunction that always yields void, used for detecting valid types.

is_detected_exact

```
template<typename _Expected , template< typename... > class _Op, typename... _Args>
using std::experimental::fundamentals\_v2::is\_detected\_exact = typedef is\_same<_Expected, detected_t<
_t<_Op, _Args...> >
```

A metafunction that always yields void, used for detecting valid types.

void_t

```
template<typename... >
using std::experimental::fundamentals\_v2::void\_t = typedef void
```

A metafunction that always yields void, used for detecting valid types.

3.15.3.4.4 Variable Documentation**is_detected_convertible_v**

```
template<typename _To , template< typename... > class _Op, typename... _Args>
constexpr bool std::experimental::fundamentals\_v2::is\_detected\_convertible\_v [constexpr]
```

A metafunction that always yields void, used for detecting valid types.

is_detected_exact_v

```
template<typename _Expected , template< typename... > class _Op, typename... _Args>
constexpr bool std::experimental::fundamentals\_v2::is\_detected\_exact\_v [constexpr]
```

A metafunction that always yields void, used for detecting valid types.

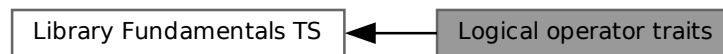
is_detected_v

```
template<template< typename... > class _Op, typename... _Args>
constexpr bool std::experimental::fundamentals_v2::is_detected_v [constexpr]
```

A metafunction that always yields void, used for detecting valid types.

3.15.3.5 Logical operator traits

Collaboration diagram for Logical operator traits:



- `#define __cpp_lib_experimental_logical_traits`
- `template<typename... _Bn>`
`constexpr bool std::experimental::conjunction_v`
- `template<typename... _Bn>`
`constexpr bool std::experimental::disjunction_v`
- `template<typename _Pp>`
`constexpr bool std::experimental::negation_v`

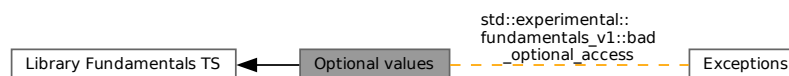
3.15.3.5.1 Detailed Description

Since

Library Fundamentals TS v2. C++14.

3.15.3.6 Optional values

Collaboration diagram for Optional values:

**Classes**

- class `std::experimental::fundamentals_v1::bad_optional_access`
- struct `std::experimental::fundamentals_v1::in_place_t`
- struct `std::experimental::fundamentals_v1::nullopt_t`
- class `std::experimental::fundamentals_v1::optional<_Tp>`

Macros

- `#define __cpp_lib_experimental_optional`

Variables

- constexpr `in_place_t` `std::experimental::in_place`
- constexpr `nullopt_t` `std::experimental::nullopt`

3.15.3.6.1 Detailed Description

Class template for optional values and surrounding facilities, as described in n3793 “A proposal to add a utility class to represent optional objects (Revision 5)”.

3.15.3.6.2 Variable Documentation

`in_place`

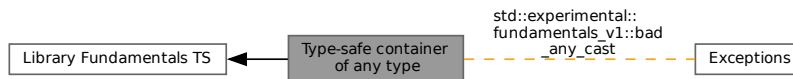
constexpr `in_place_t` `std::experimental::fundamentals_v1::in_place` [constexpr]
Tag for in-place construction.

`nullopt`

constexpr `nullopt_t` `std::experimental::fundamentals_v1::nullopt` [constexpr]
Tag to disengage optional objects.

3.15.3.7 Type-safe container of any type

Collaboration diagram for Type-safe container of any type:



Classes

- class `std::experimental::fundamentals_v1::any`
- class `std::experimental::fundamentals_v1::bad_any_cast`

Macros

- #define `__cpp_lib_experimental_any`

Functions

- static void `std::experimental::fundamentals_v1::any::Manager_internal<_Tp>::_S_manage` (`_Op` __↵ which, const `any` *__anyp, `_Arg` *__arg)
- static void `std::experimental::fundamentals_v1::any::Manager_external<_Tp>::_S_manage` (`_Op` __↵ which, const `any` *__anyp, `_Arg` *__arg)
- template<typename `_ValueType` >
 `_ValueType` `std::experimental::any_cast` (const `any` &__any)
- void `std::experimental::swap` (`any` &__x, `any` &__y) noexcept
- template<typename `_ValueType` >
 `_ValueType` `std::experimental::any_cast` (`any` &__any)

- `template<typename _ValueType , typename enable_if<!is_move_constructible< _ValueType >::value||is_lvalue_reference< _ValueType >::value, bool >::type = true>
_ValueType std::experimental::any_cast (any &&__any)`
- `template<typename _ValueType >
const _ValueType * std::experimental::any_cast (const any *__any) noexcept`
- `template<typename _ValueType >
_ValueType * std::experimental::any_cast (any *__any) noexcept`

3.15.3.7.1 Detailed Description

A type-safe container for single values of value types, as described in n3804 “Any Library Proposal (Revision 3)”.

3.15.3.7.2 Function Documentation

`any_cast()` [1/5]

```
template<typename _ValueType , typename enable\_if<!is_move_constructible< _ValueType >::value||is_lvalue_reference< _ValueType >::value, bool >::type = true>  
_ValueType std::experimental::fundamentals\_v1::any\_cast (  
    any && __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code><i>bad_any_cast</i></code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
----------------------------------	--

`any_cast()` [2/5]

```
template<typename _ValueType >  
_ValueType std::experimental::fundamentals\_v1::any\_cast (  
    any & __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code>bad_any_cast</code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------------	--

any_cast() [3/5]

```
template<typename _ValueType >
_ValueType * std::experimental::fundamentals_v1::any_cast (
    any * __any ) [inline], [noexcept]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	The type of the contained object.
-------------------------	-----------------------------------

Parameters

<code>__any</code>	A pointer to the object to access.
--------------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr && __any.type() == typeid(_ValueType)` , otherwise a null pointer.

any_cast() [4/5]

```
template<typename _ValueType >
_ValueType std::experimental::fundamentals_v1::any_cast (
    const any & __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A const-reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<i>bad_any_cast</i>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------	--

any_cast() [5/5]

```
template<typename _ValueType >
const _ValueType * std::experimental::fundamentals_v1::any_cast (
    const any * __any ) [inline], [noexcept]
```

Access the contained object.

Template Parameters

<i>_ValueType</i>	The type of the contained object.
-------------------	-----------------------------------

Parameters

<i>__any</i>	A pointer to the object to access.
--------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr && __any.type() == typeid(_ValueType)` , otherwise a null pointer.

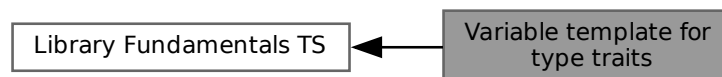
swap()

```
void std::experimental::fundamentals_v1::swap (
    any & __x,
    any & __y ) [inline], [noexcept]
```

Exchange the states of two `any` objects.

3.15.3.8 Variable template for type traits

Collaboration diagram for Variable template for type traits:



- `#define __cpp_lib_experimental_type_trait_variable_templates`

- `template<typename _Tp >`
`constexpr bool std::experimental::is_void_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_null_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_integral_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_array_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_rvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_const_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_volatile_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivial_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_standard_layout_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_empty_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_abstract_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_unsigned_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_trivially_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_default_constructible_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_nothrow_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr size_t std::experimental::alignment_of_v`
- `template<typename _Tp >`
`constexpr size_t std::experimental::rank_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::experimental::extent_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_same_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::experimental::is_base_of_v`
- `template<typename _From, typename _To >`
`constexpr bool std::experimental::is_convertible_v`

3.15.3.8.1 Detailed Description

Since

Library Fundamentals TS v1. C++14.

See also

`variable_templates`

3.15.4 Parallelism TS

Collaboration diagram for Parallelism TS:



Modules

- [Data parallel extensions](#)

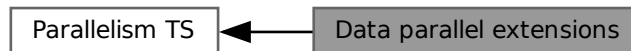
3.15.4.1 Detailed Description

Components defined by the *C++ Extensions for Parallelism* Technical Specification.

- ISO/IEC TS 19570:2015 C++ Extensions for Parallelism
- ISO/IEC TS 19570:2018 C++ Extensions for Parallelism, Version 2

3.15.4.2 Data parallel extensions

Collaboration diagram for Data parallel extensions:



Macros

- `#define __cpp_lib_experimental_parallel_simd`
- `using __m128 = float`
- `using __m128d = double`
- `using __m128i = long long`
- `using __m256 = float`
- `using __m256d = double`
- `using __m256i = long long`
- `using __m512 = float`
- `using __m512d = double`
- `using __m512i = long long`
- `template<size_t _Xp>`
`using __SizeConstant = integral_constant< size_t, _Xp >`
- `using _UChar = unsigned char`
- `using _SChar = signed char`
- `using _UShort = unsigned short`
- `using _UInt = unsigned int`
- `using _ULong = unsigned long`
- `using _ULLong = unsigned long long`
- `using _LLong = long long`
- `template<typename... _Ts>`
`using __first_of_pack_t = typename __first_of_pack< _Ts... >::type`
- `template<typename _Tp >`
`using __value_type_or_identity_t = decltype(__value_type_or_identity_impl< _Tp >(int()))`
- `template<typename _Tp, typename = enable_if_t<__is_vectorizable_v<_Tp>>>`
`using __Vectorizable = _Tp`
- `template<typename _Ptr, typename _ValueType, typename = enable_if_t< __is_possible_loadstore_conversion<_Ptr, _ValueType><←>`
`::value>>`
`using __LoadStorePtr = _Ptr`

- `template<typename _Tp >`
`using __int_for_sizeof_t = decltype(__int_for_sizeof< sizeof(_Tp)>())`
- `template<size_t _Np>`
`using __int_with_sizeof_t = decltype(__int_for_sizeof< _Np >())`
- `template<typename _Tp, typename _Up >`
`using __make_dependent_t = typename __make_dependent< _Tp, _Up >::type`
- `template<typename _Tp >`
`using __may_alias = _Tp`
- `template<size_t _Size>`
`using __bool_storage_member_type_t = typename __bool_storage_member_type< _Size >::type`
- `template<typename _Tp, int _Np>`
`using __fixed_size_storage_t = typename __fixed_size_storage< _Tp, _Np >::type`
- `template<typename _Tp >`
`using __SimdWrapper8 = _SimdWrapper< _Tp, 8/sizeof(_Tp)>`
- `template<typename _Tp >`
`using __SimdWrapper16 = _SimdWrapper< _Tp, 16/sizeof(_Tp)>`
- `template<typename _Tp >`
`using __SimdWrapper32 = _SimdWrapper< _Tp, 32/sizeof(_Tp)>`
- `template<typename _Tp >`
`using __SimdWrapper64 = _SimdWrapper< _Tp, 64/sizeof(_Tp)>`
- `template<typename _From, typename _To, typename = enable_if_t<negation< __is_narrowing_conversion<__remove_cvref_t<_From>, _To>>::value>>`
`using _ValuePreserving = _From`
- `template<typename _From, typename _To, typename _DecayedFrom = __remove_cvref_t<_From>, typename = enable_if_t<conjunction< is_convertible<_From, _To>, disjunction< is_same<_DecayedFrom, _To>, is_same<_DecayedFrom, int>, conjunction<is_same<_DecayedFrom, _UInt>, is_unsigned<_To>>, negation<__is_narrowing_conversion<_DecayedFrom, _To>>>>::value>>`
`using _ValuePreservingOrInt = _From`
- `template<typename _Tp, size_t _Size>`
`using __intrinsic_type_t = typename __intrinsic_type< _Tp, _Size *sizeof(_Tp)>::type`
- `template<typename _Tp >`
`using __intrinsic_type2_t = typename __intrinsic_type< _Tp, 2 >::type`
- `template<typename _Tp >`
`using __intrinsic_type4_t = typename __intrinsic_type< _Tp, 4 >::type`
- `template<typename _Tp >`
`using __intrinsic_type8_t = typename __intrinsic_type< _Tp, 8 >::type`
- `template<typename _Tp >`
`using __intrinsic_type16_t = typename __intrinsic_type< _Tp, 16 >::type`
- `template<typename _Tp >`
`using __intrinsic_type32_t = typename __intrinsic_type< _Tp, 32 >::type`
- `template<typename _Tp >`
`using __intrinsic_type64_t = typename __intrinsic_type< _Tp, 64 >::type`
- `template<size_t _Np>`
`using _SanitizedBitMask = _BitMask< _Np, true >`
- `template<typename _Tp, size_t _Size>`
`using __vector_type_t = typename __vector_type_n< _Tp, _Size >::type`
- `template<typename _Tp >`
`using __vector_type2_t = typename __vector_type< _Tp, 2 >::type`
- `template<typename _Tp >`
`using __vector_type4_t = typename __vector_type< _Tp, 4 >::type`
- `template<typename _Tp >`
`using __vector_type8_t = typename __vector_type< _Tp, 8 >::type`

- `template<typename _Tp >`
 `using __vector_type16_t = typename __vector_type< _Tp, 16 >::type`
- `template<typename _Tp >`
 `using __vector_type32_t = typename __vector_type< _Tp, 32 >::type`
- `template<typename _Tp >`
 `using __vector_type64_t = typename __vector_type< _Tp, 64 >::type`
- `template<typename _Tp, typename = typename _VectorTraitsImpl<_Tp>::type>`
 `using _VectorTraits = _VectorTraitsImpl< _Tp >`
- `template<typename _Tp, typename _V >`
 `using rebind_simd_t = typename rebind_simd< _Tp, _V >::type`
- `template<int _Np, typename _V >`
 `using resize_simd_t = typename resize_simd< _Np, _V >::type`
- `template<typename _Tp >`
 `using native_simd = simd< _Tp, simd_abi::native< _Tp > >`
- `template<typename _Tp, int _Np>`
 `using fixed_size_simd = simd< _Tp, simd_abi::fixed_size< _Np > >`
- `template<typename _Tp, size_t _Np>`
 `using __deduced_simd = simd< _Tp, simd_abi::deduce_t< _Tp, _Np > >`
- `template<typename _Tp >`
 `using native_simd_mask = simd_mask< _Tp, simd_abi::native< _Tp > >`
- `template<typename _Tp, int _Np>`
 `using fixed_size_simd_mask = simd_mask< _Tp, simd_abi::fixed_size< _Np > >`
- `template<typename _Tp, size_t _Np>`
 `using __deduced_simd_mask = simd_mask< _Tp, simd_abi::deduce_t< _Tp, _Np > >`
- `template<typename _Tp >`
 `using safe_make_signed_t = typename __safe_make_signed< _Tp >::type`
- `constexpr element_aligned_tag element_aligned`
- `constexpr vector_aligned_tag vector_aligned`
- `template<size_t _Np>`
 `constexpr overaligned_tag< _Np > overaligned`
- `constexpr bool __have_mmx`
- `constexpr bool __have_sse`
- `constexpr bool __have_sse2`
- `constexpr bool __have_sse3`
- `constexpr bool __have_ssse3`
- `constexpr bool __have_sse4_1`
- `constexpr bool __have_sse4_2`
- `constexpr bool __have_xop`
- `constexpr bool __have_avx`
- `constexpr bool __have_avx2`
- `constexpr bool __have_bmi`
- `constexpr bool __have_bmi2`
- `constexpr bool __have_lzcnt`
- `constexpr bool __have_sse4a`
- `constexpr bool __have_fma`
- `constexpr bool __have_fma4`
- `constexpr bool __have_f16c`
- `constexpr bool __have_popcnt`
- `constexpr bool __have_avx512f`
- `constexpr bool __have_avx512dq`
- `constexpr bool __have_avx512vl`
- `constexpr bool __have_avx512bw`

- constexpr bool **__have_avx512dq_vl**
- constexpr bool **__have_avx512bw_vl**
- constexpr bool **__have_avx512bitalg**
- constexpr bool **__have_avx512vbmi2**
- constexpr bool **__have_avx512vbmi**
- constexpr bool **__have_avx512ifma**
- constexpr bool **__have_avx512cd**
- constexpr bool **__have_avx512vnni**
- constexpr bool **__have_avx512vpopcntdq**
- constexpr bool **__have_avx512vp2intersect**
- constexpr bool **__have_neon**
- constexpr bool **__have_neon_a32**
- constexpr bool **__have_neon_a64**
- constexpr bool **__support_neon_float**
- constexpr bool **__have_power10vec**
- constexpr bool **__have_power9vec**
- constexpr bool **__have_power8vec**
- constexpr bool **__have_power_vsx**
- constexpr bool **__have_power_vmx**
- template<typename _Tp >
constexpr bool **__is_vectorizable_v**
- template<typename _Tp >
constexpr bool **__is_bitmask_v**
- template<typename _Tp >
constexpr bool **__is_fixed_size_abi_v**
- template<typename _Abi >
constexpr int **__abi_bytes_v**
- template<typename _Tp, typename _Ap >
constexpr size_t **__size_or_zero_v**
- constexpr struct _Privatelnit **__private_init**
- constexpr struct _Bitsetlnit **__bitset_init**
- template<typename _Tp >
constexpr bool **__is_simd_wrapper_v**
- template<typename _Tp = void>
static constexpr int **__min_vector_size**
- template<> constexpr int **__min_vector_size< void >**
- template<typename _Tp >
constexpr bool **__is_vector_type_v**
- template<typename _Tp >
constexpr bool **__is_intrinsic_type_v**
- template<typename _Tp >
constexpr bool **is_abi_tag_v**
- template<typename _Tp >
constexpr bool **is_simd_v**
- template<typename _Tp >
constexpr bool **is_simd_mask_v**
- template<typename _Tp, typename _Abi = simd_abi::__default_abi<_Tp>>
constexpr size_t **simd_size_v**
- template<typename _Tp, typename _Up = typename _Tp::value_type>
constexpr size_t **memory_alignment_v**

- `template<typename _Fp, size_t... _I>`
`_GLIBCXX_SIMD_INTRINSIC constexpr void __execute_on_index_sequence (_Fp &&__f, index_sequence<`
`_I... >)`
- `template<typename _Fp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr void __execute_on_index_sequence (_Fp &&, index_sequence<>)`
- `template<size_t _Np, typename _Fp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr void __execute_n_times (_Fp &&__f)`
- `template<typename _R, typename _Fp, size_t... _I>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __execute_on_index_sequence_with_return (_Fp &&__f, index_↵`
`_sequence< _I... >)`
- `template<size_t _Np, typename _R, typename _Fp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __generate_from_n_evaluations (_Fp &&__f)`
- `template<size_t... _I, typename _F0, typename _FArgs >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __call_with_n_evaluations (index_sequence< _I... >, _F0 &&↵`
`__f0, _FArgs &&__fargs)`
- `template<size_t _Np, typename _F0, typename _FArgs >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __call_with_n_evaluations (_F0 &&__f0, _FArgs &&__fargs)`
- `template<size_t _First = 0, size_t... _It, typename _Tp, typename _Fp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __call_with_subscripts (_Tp &&__x, index_sequence< _It... >, ↵`
`_Fp &&__fun)`
- `template<size_t _Np, size_t _First = 0, typename _Tp, typename _Fp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __call_with_subscripts (_Tp &&__x, _Fp &&__fun)`
- `template<typename _Tp >`
`_Tp::value_type __value_type_or_identity_impl (int)`
- `template<typename _Tp >`
`_Tp __value_type_or_identity_impl (float)`
- `template<size_t _Bytes>`
`constexpr auto __int_for_sizeof ()`
- `template<typename _Abi >`
`constexpr bool __is_scalar_abi ()`
- `template<template< int > class _Abi, int _Bytes>`
`constexpr int __abi_bytes_impl (_Abi< _Bytes > *)`
- `template<typename _Tp >`
`constexpr int __abi_bytes_impl (_Tp *)`
- `template<typename _Abi >`
`constexpr bool __is_builtin_bitmask_abi ()`
- `template<typename _Abi >`
`constexpr bool __is_sse_abi ()`
- `template<typename _Abi >`
`constexpr bool __is_avx_abi ()`
- `template<typename _Abi >`
`constexpr bool __is_avx512_abi ()`
- `template<typename _Abi >`
`constexpr bool __is_neon_abi ()`
- `template<typename... _Args>`
`_GLIBCXX_SIMD_ALWAYS_INLINE void __invoke_ub (const char *__msg, const _Args &... __args)`
- `template<typename _Tp, typename _Ap, size_t _Np = simd_size<_Tp, _Ap>::value>`
`constexpr size_t __size_or_zero_dispatch (int)`
- `template<typename _Tp, typename _Ap >`
`constexpr size_t __size_or_zero_dispatch (float)`
- `constexpr size_t __div_roundup (size_t __a, size_t __b)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC constexpr const auto & __data (const simd< _Tp, _Ap > &__x)`

- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto & __data (simd< _Tp, _Ap > &__x)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC constexpr const auto & __data (const simd_mask< _Tp, _Ap > &__x)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto & __data (simd_mask< _Tp, _Ap > &__x)`
- `template<typename _V >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __to_value_type_or_member_type (const _V &__x) ->`
`decltype(__data(__x))`
- `template<typename _V >`
`_GLIBCXX_SIMD_INTRINSIC constexpr const _V::value_type & __to_value_type_or_member_type (const`
`typename _V::value_type &__x)`
- `template<typename _V >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __as_vector (_V __x)`
- `template<size_t _Np = 0, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __as_wrapper (_V __x)`
- `template<typename _To, typename _From >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _To __intrin_bitcast (_From __v)`
- `template<typename _To, size_t _NN = 0, typename _From, typename _FromVT = _VectorTraits<_From>, size_t _Np = _NN == 0 ?`
`sizeof(_From) / sizeof(_To) : _NN>`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _To, _Np > __vector_bitcast (_From __x)`
- `template<typename _To, size_t _NN = 0, typename _Tp, size_t _Nx, size_t _Np = _NN == 0 ? sizeof(_SimdWrapper<_Tp, _Nx>) /`
`sizeof(_To) : _NN>`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _To, _Np > __vector_bitcast (const _SimdWrapper< _Tp, _Nx > &__x)`
- `template<typename _To, typename _From >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _To __bit_cast (const _From __x)`
- `template<typename _Tp, typename _TVT = _VectorTraits<_Tp>, typename _R = __intrinsic_type_t<typename _TVT::value_type, _<`
`TVT::S_full_size>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __to_intrin (_Tp __x)`
- `template<typename _Tp, typename... _Args>`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _Tp, sizeof...(_Args)> __make_vector (const _Args`
`&... __args)`
- `template<size_t _Np, typename _Tp, size_t... _I>`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _Tp, _Np > __vector_broadcast_impl (_Tp __x,`
`index_sequence< _I... >)`
- `template<size_t _Np, typename _Tp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _Tp, _Np > __vector_broadcast (_Tp __x)`
- `template<typename _Tp, size_t _Np, typename _Gp, size_t... _I>`
`_GLIBCXX_SIMD_INTRINSIC constexpr __vector_type_t< _Tp, _Np > __generate_vector_impl (_Gp &&__gen,`
`index_sequence< _I... >)`
- `template<typename _V, typename _VVT = _VectorTraits<_V>, typename _Gp >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _V __generate_vector (_Gp &&__gen)`
- `template<typename _TW >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _TW __xor (_TW __a, _TW __b) noexcept`
- `template<typename _TW >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _TW __or (_TW __a, _TW __b) noexcept`
- `template<typename _TW >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _TW __and (_TW __a, _TW __b) noexcept`
- `template<typename _TW >`
`_GLIBCXX_SIMD_INTRINSIC constexpr _TW __andnot (_TW __a, _TW __b) noexcept`
- `template<typename _Tp, typename _TVT = _VectorTraits<_Tp>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _Tp __not (_Tp __a) noexcept`

- `template<typename _Tp, typename _TVT = _VectorTraits<_Tp>, typename _R = __vector_type_t<typename _TVT::value_type, _TVT::S_full_size * 2>>`
`constexpr _R __concat (_Tp a_, _Tp b_)`
- `template<typename _Tp, typename _TVT = _VectorTraits<_Tp>>`
`_GLIBCXX_SIMD_INTRINSIC _ZeroExtendProxy<_Tp, _TVT> __zero_extend (_Tp __x)`
- `template<int _Offset, int _SplitBy, typename _Tp, typename _TVT = _VectorTraits<_Tp>, typename _R = __vector_type_t<typename _TVT::value_type, _TVT::S_full_size / _SplitBy>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __extract (_Tp __in)`
- `template<typename _Tp, typename _R = __vector_type8_t<typename _VectorTraits<_Tp>::value_type>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __lo64 (_Tp __x)`
- `template<typename _Tp, typename _R = __vector_type8_t<typename _VectorTraits<_Tp>::value_type>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __hi64 (_Tp __x)`
- `template<typename _Tp, typename _R = __vector_type8_t<typename _VectorTraits<_Tp>::value_type>>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _R __hi64z (_Tp __x)`
- `template<typename _Tp>`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __lo128 (_Tp __x)`
- `template<typename _Tp>`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __hi128 (_Tp __x)`
- `template<typename _Tp>`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __lo256 (_Tp __x)`
- `template<typename _Tp>`
`_GLIBCXX_SIMD_INTRINSIC constexpr auto __hi256 (_Tp __x)`
- `template<typename _Tp>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _AutoCast<_Tp> __auto_bitcast (const _Tp &__x)`
- `template<typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _AutoCast<typename _SimdWrapper<_Tp, _Np>::BuiltinType>`
`__auto_bitcast (const _SimdWrapper<_Tp, _Np> &__x)`
- `template<typename _Tp>`
`constexpr size_t __vectorized_sizeof ()`
- `template<typename _Tp, typename _Up, typename _Ap, typename _R = typename __static_simd_cast_return_type<_Tp, _Up, _Ap>::type>`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _R static_simd_cast (const simd<_Up, _Ap> &__x)`
- `template<typename _Tp, typename _Up, typename _Ap, typename _To = __value_type_or_identity_t<_Tp>>`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR auto simd_cast (const simd<_ValuePreserving<_Up, _To>, _Ap> &__x) -> decltype(static_simd_cast<_Tp>(__x))`
- `template<typename _Tp, int _Np>`
`_GLIBCXX_SIMD_INTRINSIC fixed_size_simd<_Tp, _Np> to_fixed_size (const fixed_size_simd<_Tp, _Np> &__x)`
- `template<typename _Tp, int _Np>`
`_GLIBCXX_SIMD_INTRINSIC fixed_size_simd_mask<_Tp, _Np> to_fixed_size (const fixed_size_simd_mask<_Tp, _Np> &__x)`
- `template<typename _Tp, typename _Ap>`
`_GLIBCXX_SIMD_INTRINSIC auto to_fixed_size (const simd<_Tp, _Ap> &__x)`
- `template<typename _Tp, typename _Ap>`
`_GLIBCXX_SIMD_INTRINSIC auto to_fixed_size (const simd_mask<_Tp, _Ap> &__x)`
- `template<typename _Tp, int _Np>`
`_GLIBCXX_SIMD_INTRINSIC enable_if_t<(_Np==native_simd<_Tp>::size()), native_simd<_Tp>> to_native (const fixed_size_simd<_Tp, _Np> &__x)`
- `template<typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC enable_if_t<(_Np==native_simd_mask<_Tp>::size()), native_simd_mask<_Tp>> to_native (const fixed_size_simd_mask<_Tp, _Np> &__x)`

- `template<typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC enable_if_t<(_Np==simd<_Tp>::size()), simd<_Tp> > to_compatible (const`
`simd<_Tp, simd_abi::fixed_size<_Np> > &__x)`
- `template<typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC enable_if_t<(_Np==simd_mask<_Tp>::size()), simd_mask<_Tp> > to_←`
`compatible (const simd_mask<_Tp, simd_abi::fixed_size<_Np> > &__x)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC where_expression< simd_mask<_Tp, _Ap>, simd<_Tp, _Ap> > where`
`(const typename simd<_Tp, _Ap>::mask_type &__k, simd<_Tp, _Ap> &__value)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC const_where_expression< simd_mask<_Tp, _Ap>, simd<_Tp, _Ap> >`
`where (const typename simd<_Tp, _Ap>::mask_type &__k, const simd<_Tp, _Ap> &__value)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC where_expression< simd_mask<_Tp, _Ap>, simd_mask<_Tp, _Ap> >`
`where (const remove_const_t< simd_mask<_Tp, _Ap> > &__k, simd_mask<_Tp, _Ap> &__value)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC const_where_expression< simd_mask<_Tp, _Ap>, simd_mask<_Tp, _Ap>`
`> where (const remove_const_t< simd_mask<_Tp, _Ap> > &__k, const simd_mask<_Tp, _Ap> &__value)`
- `template<typename _Tp >`
`_GLIBCXX_SIMD_INTRINSIC where_expression< bool, _Tp> where (_ExactBool __k, _Tp &__value)`
- `template<typename _Tp >`
`_GLIBCXX_SIMD_INTRINSIC const_where_expression< bool, _Tp> where (_ExactBool __k, const _Tp &←`
`__value)`
- `template<typename _Tp, typename _Ap >`
`void where (bool __k, simd<_Tp, _Ap> &__value)=delete`
- `template<typename _Tp, typename _Ap >`
`void where (bool __k, const simd<_Tp, _Ap> &__value)=delete`
- `template<typename _Tp, typename _Abi, typename _BinaryOperation = plus<>>`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _Tp reduce (const simd<_Tp, _Abi> &__v,`
`_BinaryOperation __binary_op=_BinaryOperation())`
- `template<typename _M, typename _V, typename _BinaryOperation = plus<>>`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &__x, type-`
`name _V::value_type __identity_element, _BinaryOperation __binary_op)`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &←`
`__x, plus<> __binary_op={})`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &←`
`__x, multiplies<> __binary_op)`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &__x, bit_←`
`and<> __binary_op)`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &__x, bit_←`
`or<> __binary_op)`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _V::value_type reduce (const const_where_expression< _M, _V> &__x, bit_←`
`xor<> __binary_op)`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _Tp hmin (const simd<_Tp, _Abi> &←`
`__v) noexcept`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _Tp hmax (const simd<_Tp, _Abi> &__v)`
`noexcept`

- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _V::value_type hmin (const const_where_↵`
`expression< _M, _V > &__x) noexcept`
- `template<typename _M, typename _V >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR _V::value_type hmax (const const_where_↵`
`expression< _M, _V > &__x) noexcept`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR simd< _Tp, _Ap > min (const simd< _Tp, _Ap`
`> &__a, const simd< _Tp, _Ap > &__b)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR simd< _Tp, _Ap > max (const simd< _Tp, _Ap`
`> &__a, const simd< _Tp, _Ap > &__b)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR pair< simd< _Tp, _Ap >, simd< _Tp, _Ap >`
`> minmax (const simd< _Tp, _Ap > &__a, const simd< _Tp, _Ap > &__b)`
- `template<typename _Tp, typename _Ap >`
`_GLIBCXX_SIMD_INTRINSIC _GLIBCXX_SIMD_CONSTEXPR simd< _Tp, _Ap > clamp (const simd< _Tp,`
`_Ap > &__v, const simd< _Tp, _Ap > &__lo, const simd< _Tp, _Ap > &__hi)`
- `template<size_t... _Sizes, typename _Tp, typename _Ap, typename = enable_if_t<((_Sizes + ...) == simd<_Tp, _Ap>::size())>>`
`tuple< simd< _Tp, simd_abi::deduce_t< _Tp, _Sizes > >... > split (const simd< _Tp, _Ap > &)`
- `template<int _Index, int _Total, int _Combine = 1, typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC _SimdWrapper< _Tp, _Np/_Total * _Combine > __extract_part (const _Simd↵`
`Wrapper< _Tp, _Np > __x)`
- `template<int _Index, int _Parts, int _Combine = 1, typename _Tp, typename _A0, typename... _As>`
`_GLIBCXX_SIMD_INTRINSIC auto __extract_part (const _SimdTUPLE< _Tp, _A0, _As... > &__x)`
- `template<typename _Tp, size_t _Np>`
`_GLIBCXX_SIMD_INTRINSIC _SimdWrapper< _Tp, _Np/2 > __extract_center (_SimdWrapper< _Tp, _Np >`
`__x)`
- `template<typename _Tp, typename _A0, typename... _As>`
`_GLIBCXX_SIMD_INTRINSIC _SimdWrapper< _Tp, _SimdTUPLE< _Tp, _A0, _As... >::S_size()/2 > ↵`
`extract_center (const _SimdTUPLE< _Tp, _A0, _As... > &__x)`
- `template<size_t... _Sizes, typename _Tp, typename... _As>`
`auto __split_wrapper (_SizeList< _Sizes... >, const _SimdTUPLE< _Tp, _As... > &__x)`
- `template<typename _V, typename _Ap, size_t _Parts = simd_size_v<typename _V::value_type, _Ap> / _V::size()`
`enable_if_t< simd_size_v< typename _V::value_type, _Ap > == _Parts * _V::size() && is_simd_v< _V >, array<`
`_V, _Parts > > split (const simd< typename _V::value_type, _Ap > &__x)`
- `template<typename _V, typename _Ap, size_t _Parts = simd_size_v<typename _V::simd_type::value_type, _Ap> / _V::size()`
`enable_if_t< is_simd_mask_v< _V > && simd_size_v< typename _V::simd_type::value_type, _Ap > == _Parts`
`* _V::size(), array< _V, _Parts > > split (const simd_mask< typename _V::simd_type::value_type, _Ap > &__x)`
- `template<size_t _I, typename _Tp, typename _Ap, typename... _As>`
`_GLIBCXX_SIMD_INTRINSIC constexpr _Tp __subscript_in_pack (const simd< _Tp, _Ap > &__x, const`
`simd< _Tp, _As > &... __xs)`
- `template<typename _Tp, typename _A0, typename... _As>`
`_GLIBCXX_SIMD_INTRINSIC void __store_pack_of_simd (char *__mem, const simd< _Tp, _A0 > &__x0,`
`const simd< _Tp, _As > &... __xs)`
- `template<typename _Tp, typename... _As, typename = __detail::__odr_helper>`
`_GLIBCXX_SIMD_CONSTEXPR simd< _Tp, simd_abi::deduce_t< _Tp, (simd_size_v< _Tp, _As > + ...) > >`
`concat (const simd< _Tp, _As > &... __xs)`
- `template<typename _Tp, typename _Abi, size_t _Np>`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR __deduced_simd< _Tp, simd_size_v<`
`_Tp, _Abi > * _Np > concat (const array< simd< _Tp, _Abi >, _Np > &__x)`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool all_of (const simd_mask< _Tp, _Abi`
`> &__k) noexcept`

- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool any_of (const simd_mask< _Tp, ↵`
`_Abi > &__k) noexcept`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool none_of (const simd_mask< _Tp,`
`_Abi > &__k) noexcept`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool some_of (const simd_mask< _Tp,`
`_Abi > &__k) noexcept`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int popcount (const simd_mask< _Tp,`
`_Abi > &__k) noexcept`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int find_first_set (const simd_mask< ↵`
`_Tp, _Abi > &__k)`
- `template<typename _Tp, typename _Abi >`
`_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int find_last_set (const simd_mask< ↵`
`_Tp, _Abi > &__k)`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool all_of (_ExactBool __x) noexcept`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool any_of (_ExactBool __x) noexcept`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool none_of (_ExactBool __x) noexcept`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR bool some_of (_ExactBool) noexcept`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int popcount (_ExactBool __x) noexcept`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int find_first_set (_ExactBool)`
- `_GLIBCXX_SIMD_ALWAYS_INLINE _GLIBCXX_SIMD_CONSTEXPR int find_last_set (_ExactBool)`

3.15.4.2.1 Detailed Description

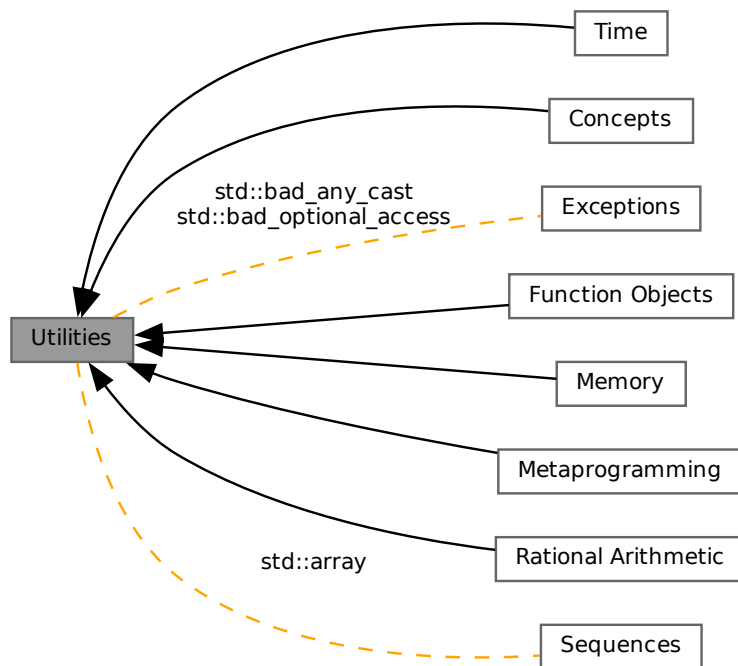
Data-parallel types library.

Since

C++17

3.16 Utilities

Collaboration diagram for Utilities:



Modules

- [Concepts](#)
- [Function Objects](#)
- [Memory](#)
- [Metaprogramming](#)
- [Rational Arithmetic](#)
- [Time](#)

Classes

- struct `std::_Optional_base<_Tp, bool, bool>`
- struct `std::_Tuple_impl<_Idx, _Elements>`
- struct `std::_Tuple_impl<_Idx, _Head, _Tail...>`
- class `std::any`
- struct `std::array<_Tp, _Nm>`
- class `std::bad_any_cast`
- class `std::bad_optional_access`

- class `std::bitset<_Nb>`
- struct `std::nullopt_t`
- class `std::optional<_Tp>`
- struct `std::pair<_T1, _T2>`
- struct `std::piecewise_construct_t`
- class `std::bitset<_Nb>::reference`
- class `std::tuple<_Elements>`
- class `std::tuple<_T1, _T2>`
- struct `std::tuple_element<_i, tuple<_Types...>>`
- struct `std::tuple_size<tuple<_Elements...>>`
- struct `std::type_index`
- struct `std::uses_allocator<tuple<_Types...>, _Alloc>`

Macros

- `#define __cpp_lib_addressof_constexpr`
- `#define __cpp_lib_any`
- `#define __cpp_lib_apply`
- `#define __cpp_lib_make_from_tuple`
- `#define __cpp_lib_optional`
- `#define __cpp_lib_tuples_by_type`

Typedefs

- `template<typename _Tp, typename _Up>`
`using std::__assigns_from_optional = __or_< is_assignable<_Tp &, const optional<_Up> &>,`
`is_assignable<_Tp &, optional<_Up> &>, is_assignable<_Tp &, const optional<_Up> &&>,`
`is_assignable<_Tp &, optional<_Up> &&>>`
- `template<typename _Tp, typename _Up>`
`using std::__converts_from_optional = __or_< is_constructible<_Tp, const optional<_Up> &>,`
`is_constructible<_Tp, optional<_Up> &>, is_constructible<_Tp, const optional<_Up> &&>,`
`is_constructible<_Tp, optional<_Up> &&>, is_convertible<const optional<_Up> &, _Tp>, is_convertible<`
`optional<_Up> &, _Tp>, is_convertible<const optional<_Up> &&, _Tp>, is_convertible<optional<_Up>`
`&&, _Tp>>`
- `template<typename _Tp>`
`using std::__empty_not_final = __conditional_t< __is_final(_Tp), false_type, __is_empty_non_tuple<_Tp>`
`>`
- `template<typename _Tp, typename _Up>`
`using std::__optional_eq_t = __optional_relop_t< decltype(std::declval<const _Tp &>())==std::declval<const`
`_Up &>()) >`
- `template<typename _Tp, typename _Up>`
`using std::__optional_ge_t = __optional_relop_t< decltype(std::declval<const _Tp &>())>=std::declval<`
`const _Up &>()) >`
- `template<typename _Tp, typename _Up>`
`using std::__optional_gt_t = __optional_relop_t< decltype(std::declval<const _Tp &>())>std::declval<const`
`_Up &>()) >`
- `template<typename _Tp, typename _Up>`
`using std::__optional_le_t = __optional_relop_t< decltype(std::declval<const _Tp &>())<=std::declval<const`
`_Up &>()) >`
- `template<typename _Tp, typename _Up>`
`using std::__optional_lt_t = __optional_relop_t< decltype(std::declval<const _Tp &>())<std::declval<const`
`_Up &>()) >`

- `template<typename _Tp, typename _Up >`
`using std::__optional_ne_t = __optional_relop_t< decltype(std::declval< const _Tp &>() !=std::declval< const`
`_Up &>()) >`
- `template<typename _Tp >`
`using std::__optional_relop_t = enable_if_t< is_convertible< _Tp, bool >::value, bool >`

Functions

- `template<typename... _Args1, typename... _Args2>`
`constexpr std::pair< _T1, _T2 >::pair (piecewise_construct_t, tuple< _Args1... >, tuple< _Args2... >)`
- `template<typename _Tp >`
`constexpr _Tp * std::__addressof (_Tp &__r) noexcept`
- `template<typename _Fn, typename _Tuple, size_t... _Idx>`
`constexpr decltype(auto) std::__apply_impl (_Fn &&__f, _Tuple &&__t, index_sequence< _Idx... >)`
- `template<typename _Tp, typename _Up = _Tp>`
`constexpr _Tp std::__exchange (_Tp &__obj, _Up &&__new_val)`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & std::__get_helper (_tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & std::__get_helper (const _tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<size_t __i, typename... _Types>`
`__enable_if_t<(__i >= sizeof...(_Types))> std::__get_helper (const tuple< _Types... > &) = delete`
- `template<typename _Tp, typename _Up = typename __inv_unwrap< _Tp >::type>`
`constexpr _Up && std::__invfwd (typename remove_reference< _Tp >::type &__t) noexcept`
- `template<typename _Callable, typename... _Args>`
`constexpr __invoke_result< _Callable, _Args... >::type std::__invoke (_Callable &&__fn, _Args &&... __args)`
`noexcept(__is_nothrow_invocable< _Callable, _Args... >::value)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`constexpr _Res std::__invoke_impl (__invoke_memfun_deref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
`args)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`constexpr _Res std::__invoke_impl (__invoke_memfun_ref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`constexpr _Res std::__invoke_impl (__invoke_memobj_deref, _MemPtr &&__f, _Tp &&__t)`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`constexpr _Res std::__invoke_impl (__invoke_memobj_ref, _MemPtr &&__f, _Tp &&__t)`
- `template<typename _Res, typename _Fn, typename... _Args>`
`constexpr _Res std::__invoke_impl (__invoke_other, _Fn &&__f, _Args &&... __args)`
- `template<typename _Res, typename _Callable, typename... _Args>`
`constexpr enable_if_t< is_invocable_r_v< _Res, _Callable, _Args... >, _Res > std::__invoke_r (_Callable`
`&&__fn, _Args &&... __args) noexcept(is_nothrow_invocable_r_v< _Res, _Callable, _Args... >)`
- `template<typename _Tp, typename _Tuple, size_t... _Idx>`
`constexpr _Tp std::__make_from_tuple_impl (_Tuple &&__t, index_sequence< _Idx... >)`
- `void std::__throw_bad_any_cast ()`
- `void std::__throw_bad_optional_access ()`
- `static void std::any::Manager_internal< _Tp >::S_manage (_Op __which, const any * __anyp, _Arg * __arg)`
- `static void std::any::Manager_external< _Tp >::S_manage (_Op __which, const any * __anyp, _Arg * __arg)`
- `template<typename _Tp >`
`constexpr _Tp * std::addressof (_Tp &__r) noexcept`
- `template<typename _Tp >`
`const _Tp * std::addressof (const _Tp &&) = delete`
- `template<typename _ValueType >`
`_ValueType std::any_cast (const any & __any)`

- `template<typename _Fn, typename _Tuple >`
`constexpr decltype(auto) std::apply (_Fn &&__f, _Tuple &&__t) noexcept(__unpack_std_tuple< is_nothrow_invocable,
_``_Fn, _Tuple >`)
- `template<typename _Tp >`
`auto std::declval () noexcept -> decltype(__declval< _Tp > (0))`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &&__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > std::forward_as_tuple (_Elements &&... __args) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp && std::get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp & std::get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp && std::get (const array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp & std::get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > && std::get (const tuple< _Elements... >
&&__t) noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > & std::get (const tuple< _Elements... >
&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp && std::get (const tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & std::get (const tuple< _Types... > &__t) noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > && std::get (tuple< _Elements... > &&__t) noex-
cept`
- `template<size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > & std::get (tuple< _Elements... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && std::get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & std::get (tuple< _Types... > &__t) noexcept`
- `template<typename _Tp, typename... _Args>`
`enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, _Args... >, any > std::make_any (_Args &&...
__args)`
- `template<typename _Tp, typename _Up, typename... _Args>`
`enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, initializer_list< _Up > &, _Args... >, any >
std::make_any (initializer_list< _Up > __il, _Args &&... __args)`
- `template<typename _Tp, typename _Tuple >`
`constexpr _Tp std::make_from_tuple (_Tuple &&__t) noexcept(__unpack_std_tuple< is_nothrow_constructible,
_Tp, _Tuple >)`
- `template<typename _Tp, typename... _Args>`
`constexpr enable_if_t< is_constructible_v< _Tp, _Args... >, optional< _Tp > > std::make_optional (_Args
&&... __args) noexcept(is_nothrow_constructible_v< _Tp, _Args... >)`
- `template<typename _Tp >`
`constexpr enable_if_t< is_constructible_v< decay_t< _Tp >, _Tp >, optional< decay_t< _Tp > > > std::
make_optional (_Tp &&__t) noexcept(is_nothrow_constructible_v< optional< decay_t< _Tp > >, _Tp >)`

- `template<typename _Tp, typename _Up, typename... _Args>`
`constexpr enable_if_t< is_constructible_v< _Tp, initializer_list< _Up > &, _Args... >, optional< _Tp > > std::`
`::make_optional (initializer_list< _Up > __il, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp,`
`initializer_list< _Up > &, _Args... >)`
- `template<typename _T1, typename _T2 >`
`constexpr pair< typename __decay_and_strip< _T1 >::__type, typename __decay_and_strip< _T2 >::__type`
`> make_pair (_T1 && __x, _T2 && __y)`
- `template<typename... _Elements>`
`constexpr tuple< typename __decay_and_strip< _Elements >::__type... > std::make_tuple (_Elements &&...`
`__args)`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && std::move (_Tp && __t) noexcept`
- `template<typename _Tp >`
`constexpr __conditional_t< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && > std::move_if_noexcept`
`(_Tp & __x) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator!= (const _Up & __lhs, const optional< _Tp > & __rhs) -> __optional_ne_t< _Up,`
`_Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator!= (const optional< _Tp > & __lhs, const _Up & __rhs) -> __optional_ne_t< _Tp,`
`_Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator!= (const optional< _Tp > & __lhs, const optional< _Up > & __rhs) -> __optional_`
`__ne_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const optional< _Tp > & __lhs, nullopt_t) noexcept`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator!= (const tuple< _TElements... > & __t, const tuple< _UElements... > & __u)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (nullopt_t, const optional< _Tp > & __rhs) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator< (const _Up & __lhs, const optional< _Tp > & __rhs) -> __optional_lt_t< _Up,`
`_Tp >`
- `template<typename _Tp >`
`constexpr bool std::operator< (const optional< _Tp > & __lhs, nullopt_t) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator< (const optional< _Tp > & __lhs, const _Up & __rhs) -> __optional_lt_t< _Tp,`
`_Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator< (const optional< _Tp > & __lhs, const optional< _Up > & __rhs) -> __optional_`
`__lt_t< _Tp, _Up >`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator< (const tuple< _TElements... > & __t, const tuple< _UElements... > & __u)`
- `template<typename _Tp >`
`constexpr bool std::operator< (nullopt_t, const optional< _Tp > & __rhs) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator<= (const _Up & __lhs, const optional< _Tp > & __rhs) -> __optional_le_t< _Up,`
`_Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator<= (const optional< _Tp > & __lhs, const _Up & __rhs) -> __optional_le_t< _Tp,`
`_Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator<= (const optional< _Tp > & __lhs, const optional< _Up > & __rhs) -> __optional_`
`__le_t< _Tp, _Up >`

- `template<typename _Tp >`
`constexpr bool std::operator<= (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp >`
`constexpr bool std::operator<= (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_eq_t< _Up, _Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator== (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp >`
`constexpr bool std::operator== (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_gt_t< _Up, _Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator> (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp >`
`constexpr bool std::operator> (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_ge_t< _Up, _Tp >`
- `template<typename _Tp >`
`constexpr bool std::operator>= (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp >`
`constexpr bool std::operator>= (nullopt_t, const optional< _Tp > &__rhs) noexcept`

- `template<typename _Tp >`
`constexpr enable_if< __and< __not< __is_tuple_like< _Tp > , is_move_constructible< _Tp > ,`
`is_move_assignable< _Tp > >::value >::type std::swap (_Tp &__a, _Tp &__b) noexcept(/*conditional */)`
`is_nothrow_move_assignable< _Tp > >`
- `template<typename _Tp, size_t _Nm>`
`constexpr enable_if< __is_swappable< _Tp >::value >::type std::swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])`
`noexcept(/*conditional */)`
- `void std::swap (any &__x, any &__y) noexcept`
- `template<typename _Tp >`
`constexpr enable_if_t< is_move_constructible_v< _Tp > &&is_swappable_v< _Tp > > std::swap (optional<`
`_Tp > &__lhs, optional< _Tp > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))`
- `template<typename... _Elements>`
`constexpr enable_if< __and< __is_swappable< _Elements >... >::value >::type std::swap (tuple< _`
`Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &... > std::tie (_Elements &... __args) noexcept`
- `template<typename... _UTypes>`
`std::tuple (_UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename... _UTypes>`
`std::tuple (allocator_arg_t, _Alloc, _UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename _T1, typename _T2 >`
`std::tuple (allocator_arg_t, _Alloc, pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename _Alloc, typename... _UTypes>`
`std::tuple (allocator_arg_t, _Alloc, tuple< _UTypes... >) -> tuple< _UTypes... >`
- `template<typename _T1, typename _T2 >`
`std::tuple (pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename... _Tpls, typename = typename enable_if<__and< __is_tuple_like< _Tpls>...>::value>::type>`
`constexpr auto std::tuple_cat (_Tpls &&... __tpls) -> typename __tuple_cat_result< _Tpls... >::__type`

Variables

- `template<typename _Tp >`
`constexpr bool std::__is_optional_v`
- `template<typename _Tp >`
`constexpr bool std::__is_optional_v< optional< _Tp > >`
- `template<template< typename... > class _Trait, typename _Tp, typename _Tuple >`
`constexpr bool std::__unpack_std_tuple`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > & >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, tuple< _Up... > & >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, tuple< _Up... > >`
- `constexpr _Swallow_assign std::ignore`
- `constexpr nullopt_t std::nullopt`
- `constexpr piecewise_construct_t std::piecewise_construct`
- `template<typename... _Types>`
`constexpr size_t std::tuple_size_v< const tuple< _Types... > >`
- `template<typename... _Types>`
`constexpr size_t std::tuple_size_v< tuple< _Types... > >`

- `template<typename _ValueType >`
`_ValueType std::any_cast (any &__any)`
- `template<typename _ValueType >`
`_ValueType std::any_cast (any &&__any)`
- `template<typename _ValueType >`
`const _ValueType * std::any_cast (const any *__any) noexcept`
- `template<typename _ValueType >`
`_ValueType * std::any_cast (any *__any) noexcept`
- `template<typename _T1 , typename _T2 >`
`pair (_T1, _T2) -> pair< _T1, _T2 >`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator== (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator< (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator!= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator> (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator<= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool operator>= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr enable_if< __and< __is_swappable< _T1 >, __is_swappable< _T2 > >::value >::type swap (pair< _T1, _T2 > &__x, pair< _T1, _T2 > &__y) noexcept(noexcept(__x.swap(__y)))`

3.16.1 Detailed Description

Basic function and class templates used with the rest of the library. Includes pair, swap, forward/move helpers, declval, integer_sequence.

3.16.2 Function Documentation

pair() [1/2]

```
template<typename _T1 , typename _T2 >
pair (
    _T1 ,
    _T2 ) -> pair< _T1, _T2 > [related]
```

Two pairs of the same type are equal iff their members are equal.

pair() [2/2]

```
template<class _T1 , class _T2 >
template<typename... _Args1, typename... _Args2>
constexpr std::pair< _T1, _T2 >::pair (
    piecewise_construct_t ,
    tuple< _Args1... > __first,
    tuple< _Args2... > __second ) [inline], [constexpr]
```

“piecewise construction” using a tuple of arguments for each member.

Parameters

<code>__first</code>	Arguments for the first member of the pair.
<code>__second</code>	Arguments for the second member of the pair.

The elements of each tuple will be used as the constructor arguments for the data members of the pair.

`__addressof()`

```
template<typename _Tp >
constexpr _Tp * std::__addressof (
    _Tp & __r ) [inline], [constexpr], [noexcept]
```

Same as C++11 `std::addressof`.

Referenced by [std::_Destroy\(\)](#), [__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#), [std::addressof\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::forward_list<_Tp, _Alloc>::merge\(\)](#), [std::list<_Tp, _Alloc>::merge\(\)](#), [std::list<_Tp, _Alloc>::merge\(\)](#), [std::sub_match<_Biter>::operator<\(\)](#), [std::sub_match<_Biter>::operator<\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=\(\)](#), [std::deque<_Tp, _Alloc>::operator=\(\)](#), [std::forward_list<_Tp, _Alloc>::operator=\(\)](#), [std::list<_Tp, _Alloc>::operator=\(\)](#), [std::vector<_Tp, _Alloc>::operator=\(\)](#), [std::sub_match<_Biter>::operator==\(\)](#), [std::sub_match<_Biter>::operator==\(\)](#), [std::rethrow_if_nested\(\)](#), [std::list<_Tp, _Alloc>::splice\(\)](#), and [std::list<_Tp, _Alloc>::splice\(\)](#).

`__invoke()`

```
template<typename _Callable , typename... _Args>
constexpr __invoke_result< _Callable, _Args... >::type std::__invoke (
    _Callable && __fn,
    _Args &&... __args ) [constexpr], [noexcept]
```

Invoke a callable object.

`addressof()`

```
template<typename _Tp >
constexpr _Tp * std::addressof (
    _Tp & __r ) [inline], [constexpr], [noexcept]
```

Returns the actual address of the object or function referenced by `r`, even in the presence of an overloaded operator`&`.

Parameters

<code>↔</code>	Reference to an object or function.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>r</code>	

Returns

The actual address.

References [std::__addressof\(\)](#).

`any_cast()` [1/5]

```
template<typename _ValueType >
_ValueType std::any_cast (
    any && __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code>bad_any_cast</code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------------	--

any_cast() [2/5]

```
template<typename _ValueType >
_ValueType std::any_cast (
    any & __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code>bad_any_cast</code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------------	--

any_cast() [3/5]

```
template<typename _ValueType >
_ValueType * std::any_cast (
    any * __any ) [inline], [noexcept]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	The type of the contained object.
-------------------------	-----------------------------------

Parameters

<code>__any</code>	A pointer to the object to access.
--------------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr && __any.type() == typeid(_ValueType)` , otherwise a null pointer.

any_cast() [4/5]

```
template<typename _ValueType >
_ValueType std::any_cast (
    const any & __any ) [inline]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	A const-reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code>bad_any_cast</code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------------	--

any_cast() [5/5]

```
template<typename _ValueType >
const _ValueType * std::any_cast (
    const any * __any ) [inline], [noexcept]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	The type of the contained object.
-------------------------	-----------------------------------

Parameters

<code>__any</code>	A pointer to the object to access.
--------------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr` && `__any.type() == typeid(decltype(ValueType))` , otherwise a null pointer.

`declval()`

```
template<typename _Tp >
auto std::declval ( ) -> decltype(__declval< _Tp > (0)) [noexcept]
```

Utility to simplify expressions used in unevaluated operands

Since

C++11

`forward()` [1/2]

```
template<typename _Tp >
constexpr _Tp && std::forward (
    typename std::remove_reference< _Tp >::type && __t ) [constexpr], [noexcept]
```

Forward an rvalue.

Returns

The parameter cast to the specified type.

This function is used to implement "perfect forwarding".

`forward()` [2/2]

```
template<typename _Tp >
constexpr _Tp && std::forward (
    typename std::remove_reference< _Tp >::type & __t ) [constexpr], [noexcept]
```

Forward an lvalue.

Returns

The parameter cast to the specified type.

This function is used to implement "perfect forwarding".

`forward_as_tuple()`

```
template<typename... _Elements>
constexpr tuple< _Elements &&... > std::forward_as_tuple (
    _Elements &&... __args ) [constexpr], [noexcept]
```

`std::forward_as_tuple`

Referenced by `std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign()`, `std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign()`, `std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace()`, and `std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace()`.

get() [1/8]

```
template<size_t __i, typename... _Elements>
constexpr const __tuple_element_t< __i, tuple< _Elements... > > && std::get (
    const tuple< _Elements... > && __t ) [constexpr], [noexcept]
```

Return a const rvalue reference to the *i*th element of a const tuple rvalue.

get() [2/8]

```
template<size_t __i, typename... _Elements>
constexpr const __tuple_element_t< __i, tuple< _Elements... > > & std::get (
    const tuple< _Elements... > & __t ) [constexpr], [noexcept]
```

Return a const reference to the *i*th element of a const tuple.

get() [3/8]

```
template<typename _Tp , typename... _Types>
constexpr const _Tp && std::get (
    const tuple< _Types... > && __t ) [constexpr], [noexcept]
```

Return a const reference to the unique element of type *_Tp* of a const tuple rvalue.

get() [4/8]

```
template<typename _Tp , typename... _Types>
constexpr const _Tp & std::get (
    const tuple< _Types... > & __t ) [constexpr], [noexcept]
```

Return a const reference to the unique element of type *_Tp* of a tuple.

get() [5/8]

```
template<size_t __i, typename... _Elements>
constexpr __tuple_element_t< __i, tuple< _Elements... > > && std::get (
    tuple< _Elements... > && __t ) [constexpr], [noexcept]
```

Return an rvalue reference to the *i*th element of a tuple rvalue.

get() [6/8]

```
template<size_t __i, typename... _Elements>
constexpr __tuple_element_t< __i, tuple< _Elements... > > & std::get (
    tuple< _Elements... > & __t ) [constexpr], [noexcept]
```

Return a reference to the *i*th element of a tuple.

get() [7/8]

```
template<typename _Tp , typename... _Types>
constexpr _Tp && std::get (
    tuple< _Types... > && __t ) [constexpr], [noexcept]
```

Return a reference to the unique element of type *_Tp* of a tuple rvalue.

get() [8/8]

```
template<typename _Tp , typename... _Types>
constexpr _Tp & std::get (
    tuple< _Types... > & __t ) [constexpr], [noexcept]
```

Return a reference to the unique element of type *_Tp* of a tuple.

make_any() [1/2]

```
template<typename _Tp , typename... _Args>
enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, _Args... >, any > std::make_any (
    _Args &&... __args ) [inline]
```

Create an any holding a `_Tp` constructed from `__args...`

make_any() [2/2]

```
template<typename _Tp , typename _Up , typename... _Args>
enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, initializer_list< _Up > &, _↵
_Args... >, any > std::make_any (
    initializer_list< _Up > __il,
    _Args &&... __args ) [inline]
```

Create an any holding a `_Tp` constructed from `__il` and `__args...`

make_pair()

```
template<typename _T1 , typename _T2 >
constexpr pair< typename __decay_and_strip< _T1 >::__type, typename __decay_and_strip< _T2 >::__↵
__type > make_pair (
    _T1 && __x,
    _T2 && __y ) [related]
```

A convenience wrapper for creating a pair from two objects.

Parameters

<code>↵ __x</code>	The first object.
<code>↵ __y</code>	The second object.

Returns

A newly-constructed `pair<>` object of the appropriate type.

The C++98 standard says the objects are passed by reference-to-const, but C++03 says they are passed by value (this was LWG issue #181).

Since C++11 they have been passed by forwarding reference and then forwarded to the new members of the pair. To create a pair with a member of reference type, pass a `reference_wrapper` to this function.

move()

```
template<typename _Tp >
constexpr std::remove_reference< _Tp >::type && std::move (
    _Tp && __t ) [constexpr], [noexcept]
```

Convert a value to an rvalue.

Parameters

<code>↵ ↵ ↵ ↵ t</code>	A thing of arbitrary type.
--	----------------------------

Returns

The parameter cast to an rvalue-reference to allow moving it.

Referenced by `std::unique_ptr<_Tp, _Dp>::~~unique_ptr()`, `std::basic_regex<_Ch_type, _Rx_traits>::assign()`, `std::shared_ptr<_Tp>::atomic_compare_exchange_strong_explicit()`, `std::shared_ptr<_Tp>::const_pointer_cast()`, `std::shared_ptr<_Tp>::dynamic_pointer_cast()`, `std::for_each_n()`, `std::get()`, `std::get()`, `std::get()`, `std::get()`, `std::get()`, `std::get()`, `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multimap<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multiset<_Key, _Compare, _Alloc>::insert()`, `std::set<_Key, _Compare, _Alloc>::insert()`, `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::vector<_Tp, _Alloc>::insert()`, `std::deque<_Tp, _Alloc>::insert()`, `std::list<_Tp, _Alloc>::insert()`, `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multimap<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::vector<_Tp, _Alloc>::insert()`, `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multimap<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multiset<_Key, _Compare, _Alloc>::insert()`, `std::set<_Key, _Compare, _Alloc>::insert()`, `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::multimap<_Key, _Tp, _Compare, _Alloc>::insert()`, `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert()`, `std::forward_list<_Tp, _Alloc>::merge()`, `std::forward_list<_Tp, _Alloc>::operator=()`, `std::move_if_noexcept()`, `std::deque<_Tp, _Alloc>::operator=()`, `std::forward_list<_Tp, _Alloc>::operator=()`, `std::function<_Res(_ArgTypes...)>::operator=()`, `std::list<_Tp, _Alloc>::operator=()`, `std::vector<_Tp, _Alloc>::operator=()`, `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::operator[]()`, `std::shared_ptr<_Tp>::reinterpret_pointer_cast()`, `std::unique_ptr<_Tp[], _Dp>::reset()`, `std::unique_ptr<_Tp, _Dp>::reset()`, `std::list<_Tp, _Alloc>::splice()`, `std::list<_Tp, _Alloc>::splice()`, and `std::shared_ptr<_Tp>::static_pointer_cast()`.

move_if_noexcept()

```
template<typename _Tp >
constexpr __conditional_t< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && > std::
::move_if_noexcept (
    _Tp & __x ) [constexpr], [noexcept]
```

Conditionally convert a value to an rvalue.

Parameters

<code>__x</code>	A thing of arbitrary type.
------------------	----------------------------

Returns

The parameter, possibly cast to an rvalue-reference.

Same as `std::move` unless the type's move constructor could throw and the type is copyable, in which case an lvalue-reference is returned instead.

References [std::move\(\)](#).

operator"!=()

```
template<typename _T1 , typename _T2 >
constexpr bool operator!= (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Uses `operator==` to find the result.

operator<()

```
template<typename _T1 , typename _T2 >
constexpr bool operator< (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Defines a lexicographical order for pairs.

For two pairs of the same type, *P* is ordered before *Q* if *P*.first is less than *Q*.first, or if *P*.first and *Q*.first are equivalent (neither is less than the other) and *P*.second is less than *Q*.second.

References [std::pair<_T1, _T2>::first](#), and [std::pair<_T1, _T2>::second](#).

operator<=()

```
template<typename _T1 , typename _T2 >
constexpr bool operator<= (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Uses `operator<` to find the result.

operator==(())

```
template<typename _T1 , typename _T2 >
constexpr bool operator== (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Two pairs of the same type are equal iff their members are equal.

References [std::pair<_T1, _T2>::first](#), and [std::pair<_T1, _T2>::second](#).

operator>()

```
template<typename _T1 , typename _T2 >
constexpr bool operator> (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Uses `operator<` to find the result.

operator>=()

```
template<typename _T1 , typename _T2 >
constexpr bool operator>= (
    const pair< _T1, _T2 > & __x,
    const pair< _T1, _T2 > & __y ) [related]
```

Uses `operator<` to find the result.

swap() [1/5]

```
template<typename _Tp >
constexpr enable_if< __and< __not< __is_tuple_like< _Tp > >, is_move_constructible< _Tp >,
is_move_assignable< _Tp > >::value >::type std::swap (
    _Tp & __a,
    _Tp & __b ) [inline], [constexpr], [noexcept]
```

Swaps two values.

Parameters

\leftrightarrow _a	A thing of arbitrary type.
\leftrightarrow _b	Another thing of arbitrary type.

Returns

Nothing.

swap() [2/5]

```
template<typename _Tp , size_t _Nm>
constexpr enable_if< __is_swappable< _Tp >::value >::type std::swap (
    _Tp(&) __a[_Nm],
    _Tp(&) __b[_Nm] ) [inline], [constexpr], [noexcept]
```

Swap the contents of two arrays.

References [std::swap\(\)](#).

swap() [3/5]

```
void std::swap (
    any & __x,
    any & __y ) [inline], [noexcept]
```

Exchange the states of two any objects.

Referenced by [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator< __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert\(\)](#), [__gnu_parallel::_LoserTree< false, _Tp, _Compare >::std::_rotate\(\)](#), [std::regex_traits< _Ch_type >::imbue\(\)](#), [std::basic_regex< _Ch_type, _Rx_traits >::imbue\(\)](#), [std::iter_swap\(\)](#), [std::swap\(\)](#), [std::basic_regex< _Ch_type, _Rx_traits >::swap\(\)](#), [std::forward_list< _Tp, _Alloc >::swap\(\)](#), [std::function< _Res\(_ArgTypes...\)>::std::match_results< _Bi_iter, _Alloc >::swap\(\)](#), [std::move_only_function< _Res\(_ArgTypes...\) _GLIBCXX_MOF_CV noexcept\(_Noex\)>::std::pair< _T1, _T2 >::swap\(\)](#), and [std::sub_match< _Biter >::swap\(\)](#).

swap() [4/5]

```
template<typename _T1 , typename _T2 >
constexpr enable_if< __and< __is_swappable< _T1 >, __is_swappable< _T2 > >::value >::type swap
(
    pair< _T1, _T2 > & __x,
    pair< _T1, _T2 > & __y ) [related]
```

Swap overload for pairs. Calls [std::pair::swap\(\)](#).

Note

This [std::swap](#) overload is not declared in C++03 mode, which has performance implications, e.g. see <https://gcc.gnu.org/PR38466>

References [std::pair< _T1, _T2 >::swap\(\)](#).

swap() [5/5]

```
template<typename... _Elements>
constexpr enable_if< __and< __is_swappable< _Elements >... >::value >::type std::swap (
    tuple< _Elements... > & __x,
    tuple< _Elements... > & __y ) [inline], [constexpr], [delete], [noexcept]
```

[swap](#)

tie()

```
template<typename... _Elements>
constexpr tuple<_Elements &...> std::tie (
    _Elements &... __args ) [constexpr], [noexcept]
```

tie

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#).**tuple_cat()**

```
template<typename... _Tpls, typename = typename enable_if<__and<__is_tuple_like<_Tpls>...>&
::value>::type>
constexpr auto std::tuple_cat (
    _Tpls &&... __tpls ) -> typename __tuple_cat_result<_Tpls...>::__type [constexpr]
```

tuple_cat

3.16.3 Variable Documentation**nullopt**

```
constexpr nullopt_t std::nullopt [inline], [constexpr]
```

Tag to disengage optional objects.

piecewise_construct

```
constexpr piecewise_construct_t std::piecewise_construct [inline], [constexpr]
```

Tag for piecewise construction of std::pair objects.

Referenced by [std::map<_Key, _Tp, _Compare, _Alloc>::insert_or_assign\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::insert_or_assign\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::operator\[\]\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::try_emplace\(\)](#), and [std::map<_Key, _Tp, _Compare, _Alloc>::try_emplace\(\)](#).

3.16.4 Concepts

Collaboration diagram for Concepts:

**Files**

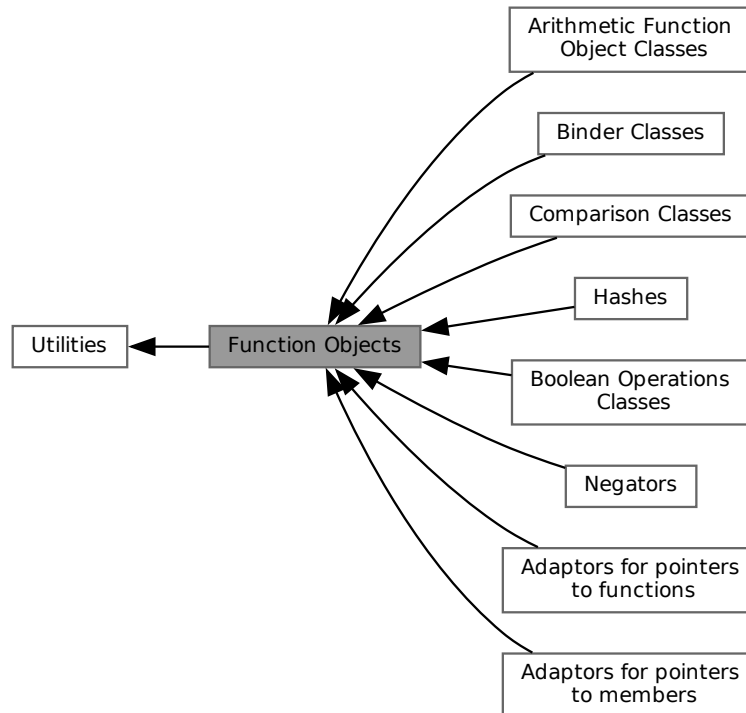
- file [concepts](#)
- file [ranges](#)

3.16.4.1 Detailed Description

Concepts for checking type requirements.

3.16.5 Function Objects

Collaboration diagram for Function Objects:



Modules

- [Adaptors for pointers to functions](#)
- [Adaptors for pointers to members](#)
- [Arithmetic Function Object Classes](#)
- [Binder Classes](#)
- [Boolean Operations Classes](#)
- [Comparison Classes](#)
- [Hashes](#)
- [Negators](#)

Classes

- `struct std::binary_function< _Arg1, _Arg2, _Result >`
- `class std::function< _Res(_ArgTypes...)>`
- `class std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>`
- `class std::reference_wrapper< _Tp >`
- `struct std::unary_function< _Arg, _Result >`

Functions

- `template<typename _Tp, typename _Class >`
`constexpr _Mem_fn< _Tp _Class::* > std::mem_fn (_Tp _Class::* __pm) noexcept`
- `template<typename _Fn >`
`constexpr auto std::not_fn (_Fn && __fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn && >::value)`

3.16.5.1 Detailed Description

Function objects, or *functors*, are objects with an `operator()` defined and accessible. They can be passed as arguments to algorithm templates and used in place of a function pointer. Not only is the resulting expressiveness of the library increased, but the generated code can be more efficient than what you might write by hand. When we refer to *functors*, then, generally we include function pointers in the description as well.

Often, functors are only created as temporaries passed to algorithm calls, rather than being created as named variables. Two examples taken from the standard itself follow. To perform a by-element addition of two vectors `a` and `b` containing

double, and put the result in `a`, use

```
transform (a.begin(), a.end(), b.begin(), a.begin(), plus<double>());
```

To negate every element in `a`, use

```
transform(a.begin(), a.end(), a.begin(), negate<double>());
```

The addition and negation functions will usually be inlined directly.

An *adaptable function object* is one which provides nested typedefs `result_type` and either `argument_type` (for a unary function) or `first_argument_type` and `second_argument_type` (for a binary function). Those typedefs are used by function object adaptors such as `bind2nd`. The standard library provides two class templates, `unary_function` and `binary_function`, which define those typedefs and so can be used as base classes of adaptable function objects.

Since C++11 the use of function object adaptors has been superseded by more powerful tools such as lambda expressions, `function<>`, and more powerful type deduction (using `auto` and `decltype`). The helpers for defining adaptable function objects are deprecated since C++11, and no longer part of the standard library since C++17. However, they are still defined and used by libstdc++ after C++17, as a conforming extension.

3.16.5.2 Function Documentation

`mem_fn()`

```
template<typename _Tp, typename _Class >
constexpr _Mem_fn< _Tp _Class::* > std::mem_fn (
    _Tp _Class::* __pm ) [inline], [constexpr], [noexcept]
```

Returns a function object that forwards to the member pointer pointer `pm`.

This allows a pointer-to-member to be transformed into a function object that can be called with an object expression as its first argument.

For a pointer-to-data-member the result must be called with exactly one argument, the object expression that would be used as the first operand in a `obj.*memptr` or `objp->*memptr` expression.

For a pointer-to-member-function the result must be called with an object expression and any additional arguments to pass to the member function, as in an expression like `(obj.*memfun)(args...)` or `(objp->*memfun)(args...)`.

The object expression can be a pointer, reference, `reference_wrapper`, or smart pointer, and the call wrapper will dereference it as needed to apply the pointer-to-member.

Since

C++11

`not_fn()`

```
template<typename _Fn >
```

```
constexpr auto std::not_fn (
    _Fn && __fn ) [inline], [constexpr], [noexcept]
```

Wrap a function object to create one that negates its result.

The function template `std::not_fn` creates a “forwarding call wrapper”, which is a function object that wraps another function object and when called, forwards its arguments to the wrapped function object.

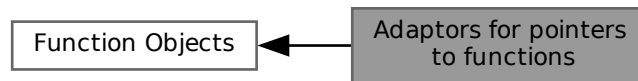
The result of invoking the wrapper is the negation (using `!`) of the wrapped function object.

Since

C++17

3.16.5.3 Adaptors for pointers to functions

Collaboration diagram for Adaptors for pointers to functions:



Classes

- class `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >`
- class `std::pointer_to_unary_function< _Arg, _Result >`

Functions

- `template<typename _Arg, typename _Result >`
`pointer_to_unary_function< _Arg, _Result > std::ptr_fun (_Result(*__x)(_Arg))`
- `template<typename _Arg1, typename _Arg2, typename _Result >`
`pointer_to_binary_function< _Arg1, _Arg2, _Result > std::ptr_fun (_Result(*__x)(_Arg1, _Arg2))`

3.16.5.3.1 Detailed Description

The advantage of function objects over pointers to functions is that the objects in the standard library declare nested typedefs describing their argument and result types with uniform names (e.g., `result_type` from the base classes `unary_function` and `binary_function`). Sometimes those typedefs are required, not just optional.

Adaptors are provided to turn pointers to unary (single-argument) and binary (double-argument) functions into function objects. The long-winded functor `pointer_to_unary_function` is constructed with a function pointer `f`, and its `operator()` called with argument `x` returns `f(x)`. The functor `pointer_to_binary_function` does the same thing, but with a double-argument `f` and `operator()`.

The function `ptr_fun` takes a pointer-to-function `f` and constructs an instance of the appropriate functor.

Deprecated Deprecated in C++11, no longer in the standard since C++17.

3.16.5.3.2 Function Documentation

`ptr_fun()` [1/2]

```
template<typename _Arg, typename _Result >
```

```
pointer_to_unary_function< _Arg, _Result > std::ptr_fun (
    _Result (*) (_Arg) __x ) [inline]
```

One of the [adaptors for function pointers](#).

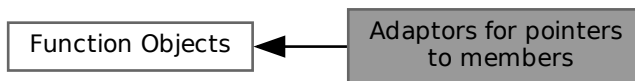
ptr_fun() [2/2]

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
pointer_to_binary_function< _Arg1, _Arg2, _Result > std::ptr_fun (
    _Result (*) (_Arg1, _Arg2) __x ) [inline]
```

One of the [adaptors for function pointers](#).

3.16.5.4 Adaptors for pointers to members

Collaboration diagram for Adaptors for pointers to members:



Classes

- class `std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >`
- class `std::const_mem_fun1_t< _Ret, _Tp, _Arg >`
- class `std::const_mem_fun_ref_t< _Ret, _Tp >`
- class `std::const_mem_fun_t< _Ret, _Tp >`
- class `std::mem_fun1_ref_t< _Ret, _Tp, _Arg >`
- class `std::mem_fun1_t< _Ret, _Tp, _Arg >`
- class `std::mem_fun_ref_t< _Ret, _Tp >`
- class `std::mem_fun_t< _Ret, _Tp >`

Functions

- template<typename _Ret, typename _Tp >
`mem_fun_t< _Ret, _Tp > std::mem_fun (_Ret(_Tp::*__f)())`
- template<typename _Ret, typename _Tp, typename _Arg >
`mem_fun1_t< _Ret, _Tp, _Arg > std::mem_fun (_Ret(_Tp::*__f)(_Arg))`
- template<typename _Ret, typename _Tp >
`mem_fun_ref_t< _Ret, _Tp > std::mem_fun_ref (_Ret(_Tp::*__f)())`
- template<typename _Ret, typename _Tp, typename _Arg >
`mem_fun1_ref_t< _Ret, _Tp, _Arg > std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg))`

3.16.5.4.1 Detailed Description

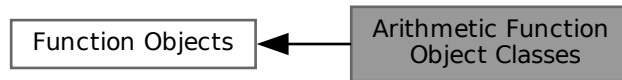
There are a total of $8 = 2^3$ function objects in this family. (1) Member functions taking no arguments vs member functions taking one argument. (2) Call through pointer vs call through reference. (3) Const vs non-const member function.

All of this complexity is in the function objects themselves. You can ignore it by using the helper function `mem_fun` and `mem_fun_ref`, which create whichever type of adaptor is appropriate.

Deprecated Deprecated in C++11, no longer in the standard since C++17. Use `mem_fn` instead.

3.16.5.5 Arithmetic Function Object Classes

Collaboration diagram for Arithmetic Function Object Classes:



Classes

- struct `std::divides<_Tp>`
- struct `std::divides<void>`
- struct `std::minus<_Tp>`
- struct `std::minus<void>`
- struct `std::modulus<_Tp>`
- struct `std::modulus<void>`
- struct `std::multiplies<_Tp>`
- struct `std::multiplies<void>`
- struct `std::negate<_Tp>`
- struct `std::negate<void>`
- struct `std::plus<_Tp>`

Macros

- `#define __cpp_lib_transparent_operators`

3.16.5.5.1 Detailed Description

The library provides function objects for basic arithmetic operations. See the documentation for [function objects](#) for examples of their use.

3.16.5.6 Binder Classes

Collaboration diagram for Binder Classes:



Namespaces

- namespace `std::placeholders`

Classes

- struct `std::_Placeholder<_Num>`
- class `std::binder1st<_Operation>`
- class `std::binder2nd<_Operation>`
- struct `std::is_bind_expression<_Tp>`
- struct `std::is_bind_expression<_Bind<_Signature>>`
- struct `std::is_bind_expression<_Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const _Bind<_Signature>>`
- struct `std::is_bind_expression<const _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<const volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_placeholder<_Tp>`
- struct `std::is_placeholder<_Placeholder<_Num>>`

Functions

- template<typename _Func, typename... _BoundArgs>
constexpr `_Bind_helper<__is_socketlike<_Func>::value, _Func, _BoundArgs...>::type` `std::bind` (`_Func` &&__f, `_BoundArgs` &&... __args)
- template<typename _Result, typename _Func, typename... _BoundArgs>
constexpr `_Bindres_helper<_Result, _Func, _BoundArgs...>::type` `std::bind` (`_Func` &&__f, `_BoundArgs` &&... __args)
- template<typename _Operation, typename _Tp>
`binder1st<_Operation>` `std::binder1st` (`const _Operation` &__fn, `const _Tp` &__x)
- template<typename _Operation, typename _Tp>
`binder2nd<_Operation>` `std::binder2nd` (`const _Operation` &__fn, `const _Tp` &__x)

3.16.5.6.1 Detailed Description

Binders turn functions/functors with two arguments into functors with a single argument, storing an argument to be applied later. For example, a variable `B` of type `binder1st` is constructed from a functor `f` and an argument `x`. Later, `B's operator()` is called with a single argument `y`. The return value is the value of `f(x, y)`. `B` can be *called* with various arguments (`y1, y2, ...`) and will in turn call `f(x, y1), f(x, y2), ...`

The function `bind1st` is provided to save some typing. It takes the function and an argument as parameters, and returns an instance of `binder1st`.

The type `binder2nd` and its creator function `bind2nd` do the same thing, but the stored argument is passed as the second parameter instead of the first, e.g., `bind2nd(std::minus<float>(), 1.3)` will create a functor whose `operator()` accepts a floating-point number, subtracts 1.3 from it, and returns the result. (If `bind1st` had been used, the functor would perform `1.3 - x` instead.

Creator-wrapper functions like `bind1st` are intended to be used in calling algorithms. Their return values will be temporary objects. (The goal is to not require you to type names like `std::binder1st<std::plus<int>>` for declaring a variable to hold the return value from `bind1st(std::plus<int>(), 5)`).

These become more useful when combined with the composition functions.

These functions are deprecated in C++11 and can be replaced by `std::bind` (or `std::tr1::bind`) which is more powerful and flexible, supporting functions with any number of arguments. Uses of `bind1st` can be replaced by `std::bind(f, x, std::placeholders::_1)` and `bind2nd` by `std::bind(f, std::placeholders::_1, x)`.

3.16.5.6.2 Function Documentation

bind() [1/2]

```
template<typename _Func , typename... _BoundArgs>
constexpr _Bind_helper< __is_socketlike< _Func >::value, _Func, _BoundArgs... >::type std::bind
(
    _Func && __f,
    _BoundArgs &&... __args ) [inline], [constexpr]
```

Function template for std::bind.

Since

C++11

bind() [2/2]

```
template<typename _Result , typename _Func , typename... _BoundArgs>
constexpr _Bindres_helper< _Result, _Func, _BoundArgs... >::type std::bind (
    _Func && __f,
    _BoundArgs &&... __args ) [inline], [constexpr]
```

Function template for std::bind<R>.

Since

C++11

bind1st()

```
template<typename _Operation , typename _Tp >
binder1st< _Operation > std::bind1st (
    const _Operation & __fn,
    const _Tp & __x ) [inline]
```

One of the [binder functors](#).

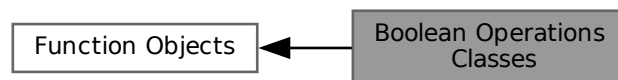
bind2nd()

```
template<typename _Operation , typename _Tp >
binder2nd< _Operation > std::bind2nd (
    const _Operation & __fn,
    const _Tp & __x ) [inline]
```

One of the [binder functors](#).

3.16.5.7 Boolean Operations Classes

Collaboration diagram for Boolean Operations Classes:



Classes

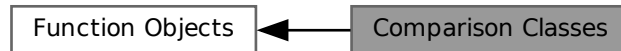
- struct [std::logical_and< _Tp >](#)
- struct [std::logical_and< void >](#)
- struct [std::logical_not< _Tp >](#)
- struct [std::logical_not< void >](#)
- struct [std::logical_or< _Tp >](#)
- struct [std::logical_or< void >](#)

3.16.5.7.1 Detailed Description

The library provides function objects for the logical operations: `&&`, `||`, and `!`.

3.16.5.8 Comparison Classes

Collaboration diagram for Comparison Classes:



Classes

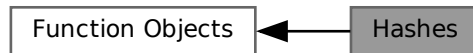
- struct [std::equal_to< _Tp >](#)
- struct [std::equal_to< void >](#)
- struct [std::greater< _Tp >](#)
- struct [std::greater< void >](#)
- struct [std::greater_equal< _Tp >](#)
- struct [std::greater_equal< void >](#)
- struct [std::less< _Tp >](#)
- struct [std::less< void >](#)
- struct [std::less_equal< _Tp >](#)
- struct [std::less_equal< void >](#)
- struct [std::not_equal_to< _Tp >](#)
- struct [std::not_equal_to< void >](#)

3.16.5.8.1 Detailed Description

The library provides six wrapper functors for all the basic comparisons in C++, like `<`.

3.16.5.9 Hashes

Collaboration diagram for Hashes:



Classes

- struct `std::hash< _Tp >`
- struct `std::hash< _Tp * >`
- struct `std::hash< bool >`
- struct `std::hash< char >`
- struct `std::hash< char16_t >`
- struct `std::hash< char32_t >`
- struct `std::hash< double >`
- struct `std::hash< float >`
- struct `std::hash< int >`
- struct `std::hash< long >`
- struct `std::hash< long double >`
- struct `std::hash< long long >`
- struct `std::hash< short >`
- struct `std::hash< signed char >`
- struct `std::hash< unsigned char >`
- struct `std::hash< unsigned int >`
- struct `std::hash< unsigned long >`
- struct `std::hash< unsigned long long >`
- struct `std::hash< unsigned short >`
- struct `std::hash< wchar_t >`

Macros

- `#define _Cxx_hashtable_define_trivial_hash(_Tp)`

3.16.5.9.1 Detailed Description

Hashing functors taking a variable type and returning a `std::size_t`.

3.16.5.10 Negators

Collaboration diagram for Negators:



Classes

- class `std::binary_negate<_Predicate>`
- class `std::unary_negate<_Predicate>`

Functions

- template<typename _Predicate>
constexpr `unary_negate<_Predicate>` `std::not1` (const _Predicate &__pred)
- template<typename _Predicate>
constexpr `binary_negate<_Predicate>` `std::not2` (const _Predicate &__pred)

3.16.5.10.1 Detailed Description

The function templates `not1` and `not2` are function object adaptors, which each take a predicate functor and wrap it in an instance of `unary_negate` or `binary_negate`, respectively. Those classes are functors whose `operator()` evaluates the wrapped predicate function and then returns the negation of the result.

For example, given a vector of integers and a trivial predicate,

```

struct IntGreaterThanThree
: public std::unary_function<int, bool>
{
    bool operator() (int x) const { return x > 3; }
};

std::find_if (v.begin(), v.end(), not1(IntGreaterThanThree()));
  
```

The call to `find_if` will locate the first index (*i*) of *v* for which `!(v[i] > 3)` is true.

The `not1/unary_negate` combination works on predicates taking a single argument. The `not2/binary_negate` combination works on predicates taking two arguments.

Deprecated Deprecated in C++17, no longer in the standard since C++20. Use `not_fn` instead.

3.16.5.10.2 Function Documentation

`not1()`

```

template<typename _Predicate>
constexpr unary_negate<_Predicate> std::not1 (
    const _Predicate & __pred ) [inline], [constexpr]
  
```

One of the [negation functors](#).

`not2()`

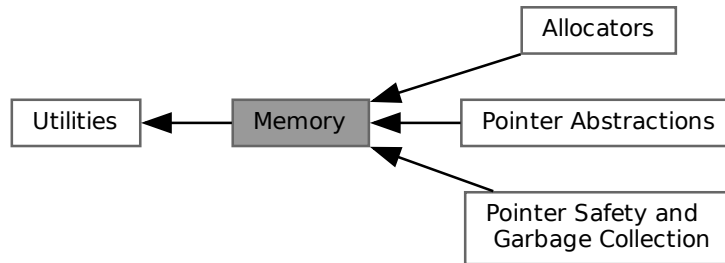
```

template<typename _Predicate>
constexpr binary_negate<_Predicate> std::not2 (
    const _Predicate & __pred ) [inline], [constexpr]
  
```

One of the [negation functors](#).

3.16.6 Memory

Collaboration diagram for Memory:



Modules

- [Allocators](#)
- [Pointer Abstractions](#)
- [Pointer Safety and Garbage Collection](#)

Files

- file [memory](#)

Macros

- `#define __cpp_lib_raw_memory_algorithms`

Functions

- `void * std::align (size_t __align, size_t __size, void *&__ptr, size_t &__space) noexcept`
- `template<size_t _Align, class _Tp >
constexpr _Tp * std::assume_aligned (_Tp *__ptr) noexcept`
- `template<typename _InputIterator, typename _ForwardIterator >
_ForwardIterator std::uninitialized_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >
_ForwardIterator std::uninitialized_copy_n (_InputIterator __first, _Size __n, _ForwardIterator __result)`
- `template<typename _ForwardIterator >
void std::uninitialized_default_construct (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Size >
_ForwardIterator std::uninitialized_default_construct_n (_ForwardIterator __first, _Size __count)`
- `template<typename _ForwardIterator, typename _Tp >
void std::uninitialized_fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp >
_ForwardIterator std::uninitialized_fill_n (_ForwardIterator __first, _Size __n, const _Tp &__x)`

- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_move (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`pair< _InputIterator, _ForwardIterator > std::uninitialized_move_n (_InputIterator __first, _Size __count, _↵`
`_ForwardIterator __result)`
- `template<typename _ForwardIterator >`
`void std::uninitialized_value_construct (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator std::uninitialized_value_construct_n (_ForwardIterator __first, _Size __count)`

3.16.6.1 Detailed Description

Components for memory allocation, deallocation, and management.

3.16.6.2 Function Documentation

align()

```
void * std::align (
    size_t __align,
    size_t __size,
    void *& __ptr,
    size_t & __space ) [inline], [noexcept]
```

Fit aligned storage in buffer.

This function tries to fit `__size` bytes of storage with alignment `__align` into the buffer `__ptr` of size `__space` bytes. If such a buffer fits then `__ptr` is changed to point to the first byte of the aligned storage and `__space` is reduced by the bytes used for alignment.

C++11 20.6.5 [ptr.align]

Parameters

<code>__align</code>	A fundamental or extended alignment value.
<code>__size</code>	Size of the aligned storage required.
<code>__ptr</code>	Pointer to a buffer of <code>__space</code> bytes.
<code>__space</code>	Size of the buffer pointed to by <code>__ptr</code> .

Returns

the updated pointer if the aligned storage fits, otherwise `nullptr`.

assume_aligned()

```
template<size_t _Align, class _Tp >
constexpr _Tp * std::assume_aligned (
    _Tp * __ptr ) [constexpr], [noexcept]
```

Inform the compiler that a pointer is aligned.

Template Parameters

<code>_Align</code>	An alignment value (i.e. a power of two)
<code>_Tp</code>	An object type

Parameters

<code>__ptr</code>	A pointer that is aligned to <code>_Align</code>
--------------------	--

C++20 20.10.6 [ptr.align]

References [std::has_single_bit\(\)](#), and [std::is_constant_evaluated\(\)](#).

uninitialized_copy()

```
template<typename _InputIterator , typename _ForwardIterator >
_FowardIterator std::uninitialized_copy (
    _InputIterator __first,
    _InputIterator __last,
    _ForwardIterator __result ) [inline]
```

Copies the range [first,last) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`__result + (__first - __last)`

Like `copy()`, but does not require an initialized output range.

Referenced by [__gnu_parallel::parallel_sort_mwms_pu\(\)](#), and [std::uninitialized_move\(\)](#).

uninitialized_copy_n()

```
template<typename _InputIterator , typename _Size , typename _ForwardIterator >
_FowardIterator std::uninitialized_copy_n (
    _InputIterator __first,
    _Size __n,
    _ForwardIterator __result ) [inline]
```

Copies the range [first,first+n) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

`__result + __n`

Since

C++11

Like `copy_n()`, but does not require an initialized output range.

References [std::__iterator_category\(\)](#).

uninitialized_default_construct()

```
template<typename _ForwardIterator >
void std::uninitialized_default_construct (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline]
```

Default-initializes objects in the range [first,last).

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Since

C++17

uninitialized_default_construct_n()

```
template<typename _ForwardIterator , typename _Size >
_FForwardIterator std::uninitialized_default_construct_n (
    _ForwardIterator __first,
    _Size __count ) [inline]
```

Default-initializes objects in the range [first,first+count).

Parameters

<code>__first</code>	A forward iterator.
<code>__count</code>	The number of objects to construct.

Returns

`__first + __count`

Since

C++17

uninitialized_fill()

```
template<typename _ForwardIterator , typename _Tp >
void std::uninitialized_fill (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Tp & __x ) [inline]
```

Copies the value x into the range [first,last).

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__x</code>	The source value.

Returns

Nothing.

Like `fill()`, but does not require an initialized output range.

uninitialized_fill_n()

```
template<typename _ForwardIterator , typename _Size , typename _Tp >
_FwdIterator std::uninitialized_fill_n (
    _ForwardIterator __first,
    _Size __n,
    const _Tp & __x ) [inline]
```

Copies the value `x` into the range `[first,first+n)`.

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of copies to make.
<code>__x</code>	The source value.

Returns

Nothing.

Like `fill_n()`, but does not require an initialized output range.

uninitialized_move()

```
template<typename _InputIterator , typename _ForwardIterator >
_FwdIterator std::uninitialized_move (
    _InputIterator __first,
    _InputIterator __last,
    _ForwardIterator __result ) [inline]
```

Move-construct from the range `[first,last)` into `result`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`__result + (__first - __last)`

Since

C++17

References [std::uninitialized_copy\(\)](#).

uninitialized_move_n()

```
template<typename _InputIterator , typename _Size , typename _ForwardIterator >
pair< _InputIterator, _ForwardIterator > std::uninitialized_move_n (
```

```

    _InputIterator __first,
    _Size __count,
    _ForwardIterator __result ) [inline]

```

Move-construct from the range [first,first+count) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__count</code>	The number of objects to initialize.
<code>__result</code>	An output iterator.

Returns

`__result + __count`

Since

C++17

uninitialized_value_construct()

```

template<typename _ForwardIterator >
void std::uninitialized_value_construct (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline]

```

Value-initializes objects in the range [first,last).

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Since

C++17

uninitialized_value_construct_n()

```

template<typename _ForwardIterator , typename _Size >
_FForwardIterator std::uninitialized_value_construct_n (
    _ForwardIterator __first,
    _Size __count ) [inline]

```

Value-initializes objects in the range [first,first+count).

Parameters

<code>__first</code>	A forward iterator.
<code>__count</code>	The number of objects to construct.

Returns

`__result + __count`

Since

C++17

3.16.6.3 Allocators

Collaboration diagram for Allocators:

**Files**

- file [scoped_allocator](#)

Classes

- struct [__gnu_cxx::__alloc_traits](#)< [_Alloc](#), [typename](#) >
- class [__gnu_cxx::__mt_alloc](#)< [_Tp](#), [_Poolp](#) >
- class [std::__new_allocator](#)< [_Tp](#) >
- class [__gnu_cxx::__pool_alloc](#)< [_Tp](#) >
- class [__gnu_cxx::__ExtPtr_allocator](#)< [_Tp](#) >
- class [std::allocator](#)< [_Tp](#) >
- class [std::allocator](#)< [void](#) >
- struct [std::allocator_traits](#)< [_Alloc](#) >
- class [__gnu_cxx::bitmap_allocator](#)< [_Tp](#) >
- class [__gnu_cxx::debug_allocator](#)< [_Alloc](#) >
- class [__gnu_cxx::malloc_allocator](#)< [_Tp](#) >
- class [__gnu_cxx::new_allocator](#)< [_Tp](#) >
- class [std::scoped_allocator_adaptor](#)< [_OuterAlloc](#), [_InnerAllocs](#) >
- class [__gnu_cxx::throw_allocator_base](#)< [_Tp](#), [_Cond](#) >
- struct [std::uses_allocator](#)< [typename](#), [typename](#) >

Macros

- `#define __cpp_lib_make_obj_using_allocator`

Typedefs

- `template<typename _Tp >`
`using std::__allocator_base = __new_allocator< _Tp >`

Functions

- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp std::make_obj_using_allocator (const _Alloc &__a, _Args &&... __args)`
- `template<typename _T1, typename _T2 >`
`constexpr bool std::operator== (const allocator< _T1 > &, const allocator< _T2 > &) noexcept`
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
`bool operator== (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_adaptor< _OutA2, _InA... > &__b) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp * std::uninitialized_construct_using_allocator (_Tp *__p, const _Alloc &__a, _Args &&... __args)`
- `template<_Std_pair _Tp, typename _Alloc >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, _Up &&, _Vp &&) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, const pair< _Up, _Vp > &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, pair< _Up, _Vp > &&) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`requires (! _Std_pair< _Tp>)`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &__a, _Args &&... __args) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Tuple1, typename _Tuple2 >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &__a, piecewise_construct_t, _Tuple1 &&__x, _Tuple2 &&__y) noexcept`

3.16.6.3.1 Detailed Description

Classes encapsulating memory operations.

3.16.6.3.2 Typedef Documentation

allocator_base

```
template<typename _Tp >
using std::__allocator_base = typedef __new_allocator<_Tp>
```

An alias to the base class for `std::allocator`.

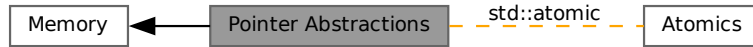
Used to set the `std::allocator` base class to `std::__new_allocator`.

Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

3.16.6.4 Pointer Abstractions

Collaboration diagram for Pointer Abstractions:



Classes

- class `std::atomic< _Tp >`
- class `std::auto_ptr< _Tp >`
- struct `std::default_delete< _Tp >`
- struct `std::default_delete< _Tp[] >`
- class `std::enable_shared_from_this< _Tp >`
- struct `std::hash< shared_ptr< _Tp > >`
- struct `std::hash< unique_ptr< _Tp, _Dp > >`
- struct `std::owner_less< _Tp >`
- struct `std::owner_less< shared_ptr< _Tp > >`
- struct `std::owner_less< void >`
- struct `std::owner_less< weak_ptr< _Tp > >`
- struct `std::pointer_traits< _Ptr >`
- struct `std::pointer_traits< _Tp * >`
- class `std::shared_ptr< _Tp >`
- class `std::unique_ptr< _Tp, _Dp >`
- class `std::unique_ptr< _Tp[], _Dp >`
- class `std::weak_ptr< _Tp >`

Macros

- `#define __cpp_lib_atomic_shared_ptr`
- `#define __cpp_lib_make_unique`

Functions

- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * std::get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<typename _Del, typename _Tp >`
`_Del * get_deleter (const shared_ptr< _Tp > &__p) noexcept`
- `template<typename _Tp, typename... _Args>`
`__GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique (_Args &&... __args)`
- `template<typename _Tp, typename... _Args>`
`__detail::__invalid_make_unique_t< _Tp > make_unique (_Args &&...)=delete`
- `template<typename _Tp >`
`__GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t< _Tp > make_unique (size_t __num)`
- `template<typename _Tp >`
`__GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique_for_overwrite ()`
- `template<typename _Tp, typename... _Args>`
`__detail::__invalid_make_unique_t< _Tp > make_unique_for_overwrite (_Args &&...)=delete`

- `template<typename _Tp >`
`_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t<_Tp> make_unique_for_overwrite (size_t __num)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool std::operator!= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator< (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _CharT, typename _Traits, typename _Tp, typename _Dp >`
`basic_ostream<_CharT, _Traits> & operator<< (basic_ostream<_CharT, _Traits> &__os, const unique_ptr<_Tp, _Dp> &__p)`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream<_Ch, _Tr> & operator<< (std::basic_ostream<_Ch, _Tr> &__os, const __shared_ptr<_Tp, _Lp> &__p)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool std::operator== (nullptr_t, const unique_ptr<_Tp, _Dp> &__x) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool std::operator> (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator> (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator> (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool std::operator>= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool std::operator>= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator>= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Del >`
`std::shared_ptr (unique_ptr<_Tp, _Del>) -> shared_ptr<_Tp>`
- `template<typename _Tp >`
`std::shared_ptr (weak_ptr<_Tp>) -> shared_ptr<_Tp>`

- `template<typename _Tp, typename _Dp >`
`enable_if<!__is_swappable< _Dp >::value >::type std::swap (unique_ptr< _Tp, _Dp > &, unique_ptr< _Tp, _Dp > &)=delete`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR enable_if< __is_swappable< _Dp >::value >::type swap (unique_ptr< _Tp, _Dp > & __x, unique_ptr< _Tp, _Dp > & __y) noexcept`
- `template<typename _Tp >`
`void swap (weak_ptr< _Tp > & __a, weak_ptr< _Tp > & __b) noexcept`
- `template<typename _Tp >`
`constexpr _Tp * std::to_address (_Tp * __ptr) noexcept`
- `template<typename _Ptr >`
`constexpr auto std::to_address (const _Ptr & __ptr) noexcept`
- `template<typename _Tp >`
`std::weak_ptr (shared_ptr< _Tp >) -> weak_ptr< _Tp >`

Variables

- `template<typename _Up >`
`static constexpr bool std::__is_shared_ptr`
- `template<typename _Up >`
`static constexpr bool std::__is_shared_ptr< shared_ptr< _Up > >`
- `template<typename _Tp, typename _Up >`
`bool operator== (const shared_ptr< _Tp > & __a, const shared_ptr< _Up > & __b) noexcept`
- `template<typename _Tp >`
`bool operator== (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator== (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator!= (const shared_ptr< _Tp > & __a, const shared_ptr< _Up > & __b) noexcept`
- `template<typename _Tp >`
`bool operator!= (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator!= (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator< (const shared_ptr< _Tp > & __a, const shared_ptr< _Up > & __b) noexcept`
- `template<typename _Tp >`
`bool operator< (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator< (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator<= (const shared_ptr< _Tp > & __a, const shared_ptr< _Up > & __b) noexcept`
- `template<typename _Tp >`
`bool operator<= (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator<= (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator> (const shared_ptr< _Tp > & __a, const shared_ptr< _Up > & __b) noexcept`
- `template<typename _Tp >`
`bool operator> (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator> (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`

- `template<typename _Tp, typename _Up >`
`bool operator>= (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept`
- `template<typename _Tp >`
`bool operator>= (const shared_ptr<_Tp> &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator>= (nullptr_t, const shared_ptr<_Tp> &__a) noexcept`
- `template<typename _Tp >`
`void swap (shared_ptr<_Tp> &__a, shared_ptr<_Tp> &__b) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> static_pointer_cast (const shared_ptr<_Up> &__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> const_pointer_cast (const shared_ptr<_Up> &__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> dynamic_pointer_cast (const shared_ptr<_Up> &__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> reinterpret_pointer_cast (const shared_ptr<_Up> &__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> static_pointer_cast (shared_ptr<_Up> &&__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> const_pointer_cast (shared_ptr<_Up> &&__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> dynamic_pointer_cast (shared_ptr<_Up> &&__r) noexcept`
- `template<typename _Tp, typename _Up >`
`shared_ptr<_Tp> reinterpret_pointer_cast (shared_ptr<_Up> &&__r) noexcept`

- `template<typename _Tp, typename _Alloc, typename... _Args>`
`shared_ptr<_NonArray<_Tp>> allocate_shared (const _Alloc &__a, _Args &&... __args)`
- `template<typename _Tp, typename... _Args>`
`shared_ptr<_NonArray<_Tp>> make_shared (_Args &&... __args)`
- `template<typename _Tp, typename _Alloc >`
`shared_ptr<_UnboundedArray<_Tp>> allocate_shared (const _Alloc &__a, size_t __n)`
- `template<typename _Tp, typename _Alloc >`
`shared_ptr<_BoundedArray<_Tp>> allocate_shared (const _Alloc &__a)`

- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_is_lock_free (const __shared_ptr<_Tp, _Lp> *__p)`

- `template<typename _Tp >`
`shared_ptr<_Tp> atomic_load_explicit (const shared_ptr<_Tp> *__p, memory_order)`

- `template<typename _Tp >`
`void atomic_store_explicit (shared_ptr<_Tp> *__p, shared_ptr<_Tp> __r, memory_order)`

- `template<typename _Tp >`
`shared_ptr<_Tp> atomic_exchange_explicit (shared_ptr<_Tp> *__p, shared_ptr<_Tp> __r, memory_order)`

- `template<typename _Tp >`
`bool atomic_compare_exchange_strong_explicit (shared_ptr<_Tp> *__p, shared_ptr<_Tp> *__w, shared_ptr<_Tp> __v, shared_ptr<_Tp> __w, memory_order, memory_order)`

3.16.6.4.1 Detailed Description

Smart pointers, etc.

3.16.6.4.2 Function Documentation

allocate_shared() [1/3]

```
template<typename _Tp , typename _Alloc >
shared_ptr< _BoundedArray< _Tp > > allocate_shared (
    const _Alloc & __a ) [related]
```

Create an object that is owned by a shared_ptr.

Parameters

<code>__a</code>	An allocator.
<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.

Returns

A shared_ptr that owns the newly created object.

Exceptions

<i>An</i>	exception thrown from <code>_Alloc::allocate</code> or from the constructor of <code>_Tp</code> .
-----------	---

A copy of `__a` will be used to allocate memory for the shared_ptr and the new object.

allocate_shared() [2/3]

```
template<typename _Tp , typename _Alloc , typename... _Args>
shared_ptr< _NonArray< _Tp > > allocate_shared (
    const _Alloc & __a,
    _Args &&... __args ) [related]
```

Create an object that is owned by a shared_ptr.

Parameters

<code>__a</code>	An allocator.
<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.

Returns

A shared_ptr that owns the newly created object.

Exceptions

<i>An</i>	exception thrown from <code>_Alloc::allocate</code> or from the constructor of <code>_Tp</code> .
-----------	---

A copy of `__a` will be used to allocate memory for the shared_ptr and the new object.

allocate_shared() [3/3]

```
template<typename _Tp , typename _Alloc >
shared_ptr< _UnboundedArray< _Tp > > allocate_shared (
    const _Alloc & __a,
```

```
size_t __n ) [related]
```

Create an object that is owned by a `shared_ptr`.

Parameters

<code>__a</code>	An allocator.
<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.

Returns

A `shared_ptr` that owns the newly created object.

Exceptions

An	exception thrown from <code>_Alloc::allocate</code> or from the constructor of <code>_Tp</code> .
----	---

A copy of `__a` will be used to allocate memory for the `shared_ptr` and the new object.

`atomic_compare_exchange_strong_explicit()`

```
template<typename _Tp >
bool atomic_compare_exchange_strong_explicit (
    shared_ptr< _Tp > * __p,
    shared_ptr< _Tp > * __v,
    shared_ptr< _Tp > __w,
    memory_order ,
    memory_order ) [related]
```

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

References [std::move\(\)](#).

`atomic_exchange_explicit()`

```
template<typename _Tp >
shared_ptr< _Tp > atomic_exchange_explicit (
    shared_ptr< _Tp > * __p,
    shared_ptr< _Tp > __r,
    memory_order ) [related]
```

Atomic exchange for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	New value to store in <code>*__p</code> .

Returns

The original value of `*__p`

References `std::shared_ptr<_Tp>::swap()`.

`atomic_is_lock_free()`

```
template<typename _Tp, _Lock_policy _Lp>
bool atomic_is_lock_free (
    const __shared_ptr<_Tp, _Lp> * __p ) [related]
```

Report whether `shared_ptr` atomic operations are lock-free.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

True if atomic access to `*__p` is lock-free, false otherwise.

`atomic_load_explicit()`

```
template<typename _Tp>
shared_ptr<_Tp> atomic_load_explicit (
    const shared_ptr<_Tp> * __p,
    memory_order ) [related]
```

Atomic load for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

`*__p`

The memory order shall not be `memory_order_release` or `memory_order_acq_rel`.

`atomic_store_explicit()`

```
template<typename _Tp>
void atomic_store_explicit (
```

```

    shared_ptr< _Tp > * __p,
    shared_ptr< _Tp > __r,
    memory_order ) [related]

```

Atomic store for shared_ptr objects.

Parameters

<code>__p</code>	A non-null pointer to a shared_ptr object.
<code>__r</code>	The value to store.

The memory order shall not be `memory_order_acquire` or `memory_order_acq_rel`.
References [`std::shared_ptr< _Tp >::swap\(\)`](#).

const_pointer_cast() [1/2]

```

template<typename _Tp , typename _Up >
shared_ptr< _Tp > const_pointer_cast (
    const shared_ptr< _Up > & __r ) [related]

```

Convert type of shared_ptr, via const_cast

const_pointer_cast() [2/2]

```

template<typename _Tp , typename _Up >
shared_ptr< _Tp > const_pointer_cast (
    shared_ptr< _Up > && __r ) [related]

```

Convert type of shared_ptr rvalue, via const_cast

Since

C++20

References [`std::move\(\)`](#).

dynamic_pointer_cast() [1/2]

```

template<typename _Tp , typename _Up >
shared_ptr< _Tp > dynamic_pointer_cast (
    const shared_ptr< _Up > & __r ) [related]

```

Convert type of shared_ptr, via dynamic_cast

dynamic_pointer_cast() [2/2]

```

template<typename _Tp , typename _Up >
shared_ptr< _Tp > dynamic_pointer_cast (
    shared_ptr< _Up > && __r ) [related]

```

Convert type of shared_ptr rvalue, via dynamic_cast

Since

C++20

References [`std::move\(\)`](#).

get_deleter()

```
template<typename _Del , typename _Tp >
_Del * get_deleter (
    const shared_ptr< _Tp > & __p ) [related]
```

20.7.2.2.10 shared_ptr get_deleter

If `__p` has a deleter of type `_Del`, return a pointer to it.

make_shared()

```
template<typename _Tp , typename... _Args>
shared_ptr< _NonArray< _Tp > > make_shared (
    _Args &&... __args ) [related]
```

Create an object that is owned by a `shared_ptr`.

Parameters

<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.
---------------------	--

Returns

A `shared_ptr` that owns the newly created object.

Exceptions

<code>std::bad_alloc</code> , or	an exception thrown from the constructor of <code>_Tp</code> .
----------------------------------	--

make_unique() [1/3]

```
template<typename _Tp , typename... _Args>
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique (
    _Args &&... __args ) [related]
```

Create an object owned by a `unique_ptr`.

Template Parameters

<code>_Tp</code>	A non-array object type.
------------------	--------------------------

Parameters

<code>__args</code>	Constructor arguments for the new object.
---------------------	---

Returns

A `unique_ptr<_Tp>` that owns the new object.

Since

C++14

make_unique() [2/3]

```
template<typename _Tp , typename... _Args>
__detail::__invalid_make_unique_t< _Tp > make_unique (
    _Args && ... ) [related]
```

Disable std::make_unique for arrays of known bound.

Template Parameters

<code>_Tp</code>	An array type of known bound, such as <code>U[N]</code> .
------------------	---

Since

C++14

make_unique() [3/3]

```
template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t< _Tp > make_unique (
    size_t __num ) [related]
```

Create an array owned by a `unique_ptr`.

Template Parameters

<code>_Tp</code>	An array type of unknown bound, such as <code>U[]</code> .
------------------	--

Parameters

<code>__num</code>	The number of elements of type <code>U</code> in the new array.
--------------------	---

Returns

A `unique_ptr<U[]>` that owns the new array.

Since

C++14

The array elements are value-initialized.

make_unique_for_overwrite() [1/3]

```
template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique_for_overwrite ( ) [related]
```

Create a default-initialised object owned by a `unique_ptr`.

Template Parameters

<code>_Tp</code>	A non-array object type.
------------------	--------------------------

Returns

A `unique_ptr<_Tp>` that owns the new object.

Since

C++20

make_unique_for_overwrite() [2/3]

```
template<typename _Tp , typename... _Args>
__detail::__invalid_make_unique_t< _Tp > make_unique_for_overwrite (
    _Args && ... ) [related]
```

Disable `std::make_unique_for_overwrite` for arrays of known bound.

Template Parameters

<code>_Tp</code>	An array type of known bound, such as <code>U[N]</code> .
------------------	---

Since

C++20

References [std::unique_ptr<_Tp, _Dp>::get\(\)](#).

make_unique_for_overwrite() [3/3]

```
template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t< _Tp > make_unique_for_overwrite (
    size_t __num ) [related]
```

Create a default-initialized array owned by a `unique_ptr`.

Template Parameters

<code>_Tp</code>	An array type of unknown bound, such as <code>U[]</code> .
------------------	--

Parameters

<code>__num</code>	The number of elements of type <code>U</code> in the new array.
--------------------	---

Returns

A `unique_ptr<U[]>` that owns the new array.

Since

C++20

operator!=() [1/6]

```
template<typename _Tp , typename _Up >
bool operator!= (
```



```
const shared_ptr< _Tp > & __a,
const shared_ptr< _Up > & __b ) [related]
```

Inequality operator for shared_ptr objects, compares the stored pointers.

operator"!="() [2/6]

```
template<typename _Tp >
bool operator!= (
    const shared_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

shared_ptr comparison with nullptr

operator"!="() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
bool std::operator!= (
    const unique_ptr< _Tp, _Dp > & __x,
    const unique_ptr< _Up, _Ep > & __y ) [inline]
```

Inequality operator for unique_ptr objects, compares the owned pointers.

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator"!="() [4/6]

```
template<typename _Tp , typename _Dp >
bool std::operator!= (
    const unique_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline], [noexcept]
```

unique_ptr comparison with nullptr

operator"!="() [5/6]

```
template<typename _Tp >
bool operator!= (
    nullptr_t ,
    const shared_ptr< _Tp > & __a ) [related]
```

shared_ptr comparison with nullptr

operator"!="() [6/6]

```
template<typename _Tp , typename _Dp >
bool std::operator!= (
    nullptr_t ,
    const unique_ptr< _Tp, _Dp > & __x ) [inline], [noexcept]
```

unique_ptr comparison with nullptr

operator<() [1/6]

```
template<typename _Tp , typename _Up >
bool operator< (
    const shared_ptr< _Tp > & __a,
    const shared_ptr< _Up > & __b ) [related]
```

Relational operator for shared_ptr objects, compares the stored pointers.

operator<>() [2/6]

```
template<typename _Tp >
bool operator< (
    const shared\_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator<>() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR bool std::operator< (
    const unique\_ptr< _Tp, _Dp > & __x,
    const unique\_ptr< _Up, _Ep > & __y ) [inline]
```

Relational operator for [unique_ptr](#) objects, compares the owned pointers.

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator<>() [4/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator< (
    const unique\_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline]
```

[unique_ptr](#) comparison with [nullptr](#)

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator<>() [5/6]

```
template<typename _Tp >
bool operator< (
    nullptr_t ,
    const shared\_ptr< _Tp > & __a ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator<>() [6/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator< (
    nullptr_t ,
    const unique\_ptr< _Tp, _Dp > & __x ) [inline]
```

[unique_ptr](#) comparison with [nullptr](#)

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator<<>() [1/2]

```
template<typename _CharT , typename _Traits , typename _Tp , typename _Dp >
basic\_ostream< _CharT, _Traits > & operator<< (
    basic\_ostream< _CharT, _Traits > & __os,
    const unique\_ptr< _Tp, _Dp > & __p ) [related]
```

Stream output operator for [unique_ptr](#).

Since

C++20

operator<<() [2/2]

```
template<typename _Ch , typename _Tr , typename _Tp , _Lock_policy _Lp>
std::basic_ostream< _Ch, _Tr > & operator<< (
    std::basic_ostream< _Ch, _Tr > & __os,
    const __shared_ptr< _Tp, _Lp > & __p ) [related]
```

Write the stored pointer to an ostream.

References [std::operator<<\(\)](#).

operator<=() [1/6]

```
template<typename _Tp , typename _Up >
bool operator<= (
    const shared_ptr< _Tp > & __a,
    const shared_ptr< _Up > & __b ) [related]
```

Relational operator for shared_ptr objects, compares the stored pointers.

operator<=() [2/6]

```
template<typename _Tp >
bool operator<= (
    const shared_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

shared_ptr comparison with nullptr

operator<=() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR bool std::operator<= (
    const unique_ptr< _Tp, _Dp > & __x,
    const unique_ptr< _Up, _Ep > & __y ) [inline]
```

Relational operator for unique_ptr objects, compares the owned pointers.

operator<=() [4/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator<= (
    const unique_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline]
```

unique_ptr comparison with nullptr

operator<=() [5/6]

```
template<typename _Tp >
bool operator<= (
    nullptr_t ,
    const shared_ptr< _Tp > & __a ) [related]
```

shared_ptr comparison with nullptr

operator<=() [6/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator<= (
    nullptr_t ,
    const unique_ptr< _Tp, _Dp > & __x ) [inline]
```

unique_ptr comparison with nullptr

operator==() [1/6]

```
template<typename _Tp , typename _Up >
bool operator== (
    const shared\_ptr< _Tp > & __a,
    const shared\_ptr< _Up > & __b ) [related]
```

Equality operator for [shared_ptr](#) objects, compares the stored pointers

operator==() [2/6]

```
template<typename _Tp >
bool operator== (
    const shared\_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator==() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR bool std::operator== (
    const unique\_ptr< _Tp, _Dp > & __x,
    const unique\_ptr< _Up, _Ep > & __y ) [inline]
```

Equality operator for [unique_ptr](#) objects, compares the owned pointers.

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator==() [4/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator== (
    const unique\_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline], [noexcept]
```

[unique_ptr](#) comparison with [nullptr](#)

operator==() [5/6]

```
template<typename _Tp >
bool operator== (
    nullptr_t ,
    const shared\_ptr< _Tp > & __a ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator==() [6/6]

```
template<typename _Tp , typename _Dp >
bool std::operator== (
    nullptr_t ,
    const unique\_ptr< _Tp, _Dp > & __x ) [inline], [noexcept]
```

[unique_ptr](#) comparison with [nullptr](#)

operator>() [1/6]

```
template<typename _Tp , typename _Up >
bool operator> (
    const shared\_ptr< _Tp > & __a,
    const shared\_ptr< _Up > & __b ) [related]
```

Relational operator for [shared_ptr](#) objects, compares the stored pointers.

operator>() [2/6]

```
template<typename _Tp >
bool operator> (
    const shared\_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator>() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR bool std::operator> (
    const unique\_ptr< _Tp, _Dp > & __x,
    const unique\_ptr< _Up, _Ep > & __y ) [inline]
```

Relational operator for [unique_ptr](#) objects, compares the owned pointers.

operator>() [4/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator> (
    const unique\_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline]
```

[unique_ptr](#) comparison with [nullptr](#)

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator>() [5/6]

```
template<typename _Tp >
bool operator> (
    nullptr_t ,
    const shared\_ptr< _Tp > & __a ) [related]
```

[shared_ptr](#) comparison with [nullptr](#)

operator>() [6/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator> (
    nullptr_t ,
    const unique\_ptr< _Tp, _Dp > & __x ) [inline]
```

[unique_ptr](#) comparison with [nullptr](#)

References [std::unique_ptr< _Tp, _Dp >::get\(\)](#).

operator>=() [1/6]

```
template<typename _Tp , typename _Up >
bool operator>= (
    const shared\_ptr< _Tp > & __a,
    const shared\_ptr< _Up > & __b ) [related]
```

Relational operator for [shared_ptr](#) objects, compares the stored pointers.

operator>=() [2/6]

```
template<typename _Tp >
bool operator>= (
    const shared\_ptr< _Tp > & __a,
    nullptr_t ) [related]
```

shared_ptr comparison with nullptr

operator>=() [3/6]

```
template<typename _Tp , typename _Dp , typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR bool std::operator>= (
    const unique_ptr< _Tp, _Dp > & __x,
    const unique_ptr< _Up, _Ep > & __y ) [inline]
```

Relational operator for unique_ptr objects, compares the owned pointers.

operator>=() [4/6]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR bool std::operator>= (
    const unique_ptr< _Tp, _Dp > & __x,
    nullptr_t ) [inline]
```

unique_ptr comparison with nullptr

operator>=() [5/6]

```
template<typename _Tp >
bool operator>= (
    nullptr_t ,
    const shared_ptr< _Tp > & __a ) [related]
```

shared_ptr comparison with nullptr

operator>=() [6/6]

```
template<typename _Tp , typename _Dp >
bool std::operator>= (
    nullptr_t ,
    const unique_ptr< _Tp, _Dp > & __x ) [inline]
```

unique_ptr comparison with nullptr

reinterpret_pointer_cast() [1/2]

```
template<typename _Tp , typename _Up >
shared_ptr< _Tp > reinterpret_pointer_cast (
    const shared_ptr< _Up > & __r ) [related]
```

Convert type of shared_ptr, via reinterpret_cast

Since

C++17

reinterpret_pointer_cast() [2/2]

```
template<typename _Tp , typename _Up >
shared_ptr< _Tp > reinterpret_pointer_cast (
    shared_ptr< _Up > && __r ) [related]
```

Convert type of shared_ptr rvalue, via reinterpret_cast

Since

C++20

References [std::move\(\)](#).

static_pointer_cast() [1/2]

```
template<typename _Tp , typename _Up >
shared_ptr< _Tp > static_pointer_cast (
    const shared_ptr< _Up > & __r ) [related]
```

Convert type of `shared_ptr`, via `static_cast`

static_pointer_cast() [2/2]

```
template<typename _Tp , typename _Up >
shared_ptr< _Tp > static_pointer_cast (
    shared_ptr< _Up > && __r ) [related]
```

Convert type of `shared_ptr` rvalue, via `static_cast`

Since

C++20

References [std::move\(\)](#).

swap() [1/3]

```
template<typename _Tp >
void swap (
    shared_ptr< _Tp > & __a,
    shared_ptr< _Tp > & __b ) [related]
```

Swap overload for `shared_ptr`.

Referenced by [std::shared_ptr< _Tp >::atomic_exchange_explicit\(\)](#), and [std::shared_ptr< _Tp >::atomic_store_explicit\(\)](#).

swap() [2/3]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR enable_if< __is_swappable< _Dp >::value >::type swap (
    unique_ptr< _Tp, _Dp > & __x,
    unique_ptr< _Tp, _Dp > & __y ) [related]
```

Swap overload for `unique_ptr`

swap() [3/3]

```
template<typename _Tp >
void swap (
    weak_ptr< _Tp > & __a,
    weak_ptr< _Tp > & __b ) [related]
```

Swap overload for `weak_ptr`.

to_address() [1/2]

```
template<typename _Tp >
constexpr _Tp * std::to_address (
    _Tp * __ptr ) [constexpr], [noexcept]
```

Obtain address referenced by a pointer to an object.

Parameters

<code>__ptr</code>	A pointer to an object
--------------------	------------------------

Returns`__ptr`**to_address() [2/2]**

```
template<typename _Ptr >
constexpr auto std::to_address (
    const _Ptr & __ptr ) [constexpr], [noexcept]
```

Obtain address referenced by a pointer to an object.

Parameters

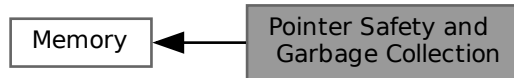
<code>__ptr</code>	A pointer to an object
--------------------	------------------------

Returns

`pointer_traits<_Ptr>::to_address(__ptr)` if that expression is well-formed, otherwise `to_address(__ptr.operator->())`

3.16.6.5 Pointer Safety and Garbage Collection

Collaboration diagram for Pointer Safety and Garbage Collection:

**Enumerations**

- enum class `std::pointer_safety` { `relaxed` , `preferred` , `strict` }

Functions

- void `std::declare_no_pointers` (char *, size_t)
- void `std::declare_reachable` (void *)
- `pointer_safety` `std::get_pointer_safety` () noexcept
- void `std::undeclare_no_pointers` (char *, size_t)
- template<typename _Tp >
_Tp * `std::undeclare_reachable` (_Tp * __p)

3.16.6.5.1 Detailed Description

Utilities to assist with garbage collection in an implementation that supports *strict pointer safety*. This implementation only supports *relaxed pointer safety* and so these functions have no effect.

C++11 20.6.4 [util.dynamic.safety], Pointer safety

3.16.6.5.2 Enumeration Type Documentation

pointer_safety

enum class `std::pointer_safety` [strong]

Constants representing the different types of pointer safety.

3.16.6.5.3 Function Documentation

declare_no_pointers()

```
void std::declare_no_pointers (
    char * ,
    size_t ) [inline]
```

Inform a garbage collector that a region of memory need not be traced.

declare_reachable()

```
void std::declare_reachable (
    void * ) [inline]
```

Inform a garbage collector that an object is still in use.

get_pointer_safety()

```
pointer_safety std::get_pointer_safety ( ) [inline], [noexcept]
```

The type of pointer safety supported by the implementation.

undeclare_no_pointers()

```
void std::undeclare_no_pointers (
    char * ,
    size_t ) [inline]
```

Unregister a range previously registered with `declare_no_pointers`.

undeclare_reachable()

```
template<typename _Tp >
_Tp * std::undeclare_reachable (
    _Tp * __p ) [inline]
```

Unregister an object previously registered with `declare_reachable`.

3.16.7 Metaprogramming

Collaboration diagram for Metaprogramming:



Modules

- [Variable templates for type traits](#)

Classes

- [struct `std::__is_nullptr_t<_Tp>`](#)
- [struct `std::tr2::__reflection_typelist<_Elements>`](#)
- [struct `std::tr2::__reflection_typelist<_First, _Rest...>`](#)
- [struct `std::tr2::__reflection_typelist<>`](#)
- [struct `std::add_const<_Tp>`](#)
- [struct `std::add_cv<_Tp>`](#)
- [struct `std::add_lvalue_reference<_Tp>`](#)
- [struct `std::add_pointer<_Tp>`](#)
- [struct `std::add_rvalue_reference<_Tp>`](#)
- [struct `std::add_volatile<_Tp>`](#)
- [struct `std::aligned_storage<_Len, _Align>`](#)
- [struct `std::aligned_union<_Len, _Types>`](#)
- [struct `std::alignment_of<_Tp>`](#)
- [struct `std::tr2::bases<_Tp>`](#)
- [struct `std::common_type<_Tp>`](#)
- [struct `std::conditional<_Cond, _Iftrue, _Iffalse>`](#)
- [class `std::decay<_Tp>`](#)
- [struct `std::tr2::direct_bases<_Tp>`](#)
- [struct `std::enable_if<bool, _Tp>`](#)
- [struct `std::extent<typename, _Uint>`](#)
- [struct `std::has_unique_object_representations<_Tp>`](#)
- [struct `std::has_virtual_destructor<_Tp>`](#)
- [struct `std::integral_constant<_Tp, __v>`](#)
- [struct `std::invoke_result<_Functor, _ArgTypes>`](#)
- [struct `std::is_abstract<_Tp>`](#)
- [struct `std::is_aggregate<_Tp>`](#)
- [struct `std::is_arithmetic<_Tp>`](#)
- [struct `std::is_array<typename>`](#)
- [struct `std::is_assignable<_Tp, _Up>`](#)
- [struct `std::is_base_of<_Base, _Derived>`](#)
- [struct `std::is_bounded_array<_Tp>`](#)
- [struct `std::is_class<_Tp>`](#)
- [struct `std::is_compound<_Tp>`](#)
- [struct `std::is_const<typename>`](#)
- [struct `std::is_constructible<_Tp, _Args>`](#)
- [struct `std::is_convertible<_From, _To>`](#)
- [struct `std::is_copy_assignable<_Tp>`](#)
- [struct `std::is_copy_constructible<_Tp>`](#)
- [struct `std::is_default_constructible<_Tp>`](#)
- [struct `std::is_destructible<_Tp>`](#)
- [struct `std::is_empty<_Tp>`](#)
- [struct `std::is_enum<_Tp>`](#)
- [struct `std::is_final<_Tp>`](#)
- [struct `std::is_floating_point<_Tp>`](#)
- [struct `std::is_function<_Tp>`](#)
- [struct `std::is_fundamental<_Tp>`](#)

- struct `std::is_integral< _Tp >`
- struct `std::is_invocable< _Fn, _ArgTypes >`
- struct `std::is_invocable_r< _Ret, _Fn, _ArgTypes >`
- struct `std::is_layout_compatible< _Tp, _Up >`
- struct `std::is_literal_type< _Tp >`
- struct `std::is_lvalue_reference< typename >`
- struct `std::is_member_function_pointer< _Tp >`
- struct `std::is_member_object_pointer< _Tp >`
- struct `std::is_member_pointer< _Tp >`
- struct `std::is_move_assignable< _Tp >`
- struct `std::is_move_constructible< _Tp >`
- struct `std::is_nothrow_assignable< _Tp, _Up >`
- struct `std::is_nothrow_constructible< _Tp, _Args >`
- struct `std::is_nothrow_convertible< _From, _To >`
- struct `std::is_nothrow_copy_assignable< _Tp >`
- struct `std::is_nothrow_copy_constructible< _Tp >`
- struct `std::is_nothrow_default_constructible< _Tp >`
- struct `std::is_nothrow_destructible< _Tp >`
- struct `std::is_nothrow_invocable< _Fn, _ArgTypes >`
- struct `std::is_nothrow_invocable_r< _Ret, _Fn, _ArgTypes >`
- struct `std::is_nothrow_move_assignable< _Tp >`
- struct `std::is_nothrow_move_constructible< _Tp >`
- struct `std::is_nothrow_swappable< _Tp >`
- struct `std::is_nothrow_swappable_with< _Tp, _Up >`
- struct `std::is_null_pointer< _Tp >`
- struct `std::is_object< _Tp >`
- struct `std::is_pod< _Tp >`
- struct `std::is_pointer< _Tp >`
- struct `std::is_pointer_interconvertible_base_of< _Base, _Derived >`
- struct `std::is_polymorphic< _Tp >`
- struct `std::is_reference< _Tp >`
- struct `std::is_rvalue_reference< typename >`
- struct `std::is_same< _Tp, _Up >`
- struct `std::is_scalar< _Tp >`
- struct `std::is_signed< _Tp >`
- struct `std::is_standard_layout< _Tp >`
- struct `std::is_swappable< _Tp >`
- struct `std::is_swappable_with< _Tp, _Up >`
- struct `std::is_trivial< _Tp >`
- struct `std::is_trivially_assignable< _Tp, _Up >`
- struct `std::is_trivially_constructible< _Tp, _Args >`
- struct `std::is_trivially_copy_assignable< _Tp >`
- struct `std::is_trivially_copy_constructible< _Tp >`
- struct `std::is_trivially_copyable< _Tp >`
- struct `std::is_trivially_default_constructible< _Tp >`
- struct `std::is_trivially_destructible< _Tp >`
- struct `std::is_trivially_move_assignable< _Tp >`
- struct `std::is_trivially_move_constructible< _Tp >`
- struct `std::is_unbounded_array< _Tp >`
- struct `std::is_union< _Tp >`
- struct `std::is_unsigned< _Tp >`

- struct `std::is_void< _Tp >`
- struct `std::is_volatile< typename >`
- struct `std::make_signed< _Tp >`
- struct `std::make_unsigned< _Tp >`
- struct `std::rank< typename >`
- struct `std::remove_all_extents< _Tp >`
- struct `std::remove_const< _Tp >`
- struct `std::remove_cv< _Tp >`
- struct `std::remove_extent< _Tp >`
- struct `std::remove_pointer< _Tp >`
- struct `std::remove_reference< _Tp >`
- struct `std::remove_volatile< _Tp >`
- struct `std::result_of< _Signature >`
- struct `std::underlying_type< _Tp >`
- struct `std::unwrap_ref_decay< _Tp >`
- struct `std::unwrap_reference< _Tp >`

Macros

- `#define __cpp_lib_bool_constant`
- `#define __cpp_lib_bounded_array_traits`
- `#define __cpp_lib_has_unique_object_representations`
- `#define __cpp_lib_is_aggregate`
- `#define __cpp_lib_is_constant_evaluated`
- `#define __cpp_lib_is_final`
- `#define __cpp_lib_is_invocable`
- `#define __cpp_lib_is_layout_compatible`
- `#define __cpp_lib_is_nothrow_convertible`
- `#define __cpp_lib_is_null_pointer`
- `#define __cpp_lib_is_pointer_interconvertible`
- `#define __cpp_lib_is_swappable`
- `#define __cpp_lib_logical_traits`
- `#define __cpp_lib_result_of_sfinae`
- `#define __cpp_lib_transformation_trait_aliases`
- `#define __cpp_lib_type_trait_variable_templates`
- `#define __cpp_lib_unwrap_ref`
- `#define __cpp_lib_void_t`

Typedefs

- `template<bool _Cond, typename _If, typename _Else >`
`using std::__conditional_t = typename __conditional< _Cond >::template type< _If, _Else >`
- `template<typename _ToElementType, typename _FromElementType >`
`using std::__is_array_convertible = is_convertible< _FromElementType(*)[], _ToElementType(*)[]>`
- `template<typename _Tp, typename _Up >`
`using std::__is_nothrow_assignable_impl = __bool_constant< __is_nothrow_assignable(_Tp, _Up)>`
- `template<typename _Tp >`
`using std::add_const_t = typename add_const< _Tp >::type`
- `template<typename _Tp >`
`using std::add_cv_t = typename add_cv< _Tp >::type`
- `template<typename _Tp >`
`using std::add_lvalue_reference_t = typename add_lvalue_reference< _Tp >::type`

- `template<typename _Tp >`
`using std::add_pointer_t = typename add_pointer< _Tp >::type`
- `template<typename _Tp >`
`using std::add_rvalue_reference_t = typename add_rvalue_reference< _Tp >::type`
- `template<typename _Tp >`
`using std::add_volatile_t = typename add_volatile< _Tp >::type`
- `template<size_t _Len, size_t _Align = __alignof__(typename __aligned_storage_msa<_Len>::__type)>`
`using std::aligned_storage_t = typename aligned_storage< _Len, _Align >::type`
- `template<size_t _Len, typename... _Types>`
`using std::aligned_union_t = typename aligned_union< _Len, _Types... >::type`
- `template<bool __v>`
`using std::bool_constant = integral_constant< bool, __v >`
- `template<typename... _Tp>`
`using std::common_reference_t = typename common_reference< _Tp... >::type`
- `template<typename... _Tp>`
`using std::common_type_t = typename common_type< _Tp... >::type`
- `template<bool _Cond, typename _Iftrue, typename _Iffalse >`
`using std::conditional_t = typename conditional< _Cond, _Iftrue, _Iffalse >::type`
- `template<typename _Tp >`
`using std::decay_t = typename decay< _Tp >::type`
- `template<bool _Cond, typename _Tp = void>`
`using std::enable_if_t = typename enable_if< _Cond, _Tp >::type`
- `using std::false_type = integral_constant< bool, false >`
- `template<typename _Fn, typename... _Args>`
`using std::invoke_result_t = typename invoke_result< _Fn, _Args... >::type`
- `template<typename _Tp >`
`using std::make_signed_t = typename make_signed< _Tp >::type`
- `template<typename _Tp >`
`using std::make_unsigned_t = typename make_unsigned< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_all_extents_t = typename remove_all_extents< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_const_t = typename remove_const< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_cv_t = typename remove_cv< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_extent_t = typename remove_extent< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_pointer_t = typename remove_pointer< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_reference_t = typename remove_reference< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_volatile_t = typename remove_volatile< _Tp >::type`
- `template<typename _Tp >`
`using std::result_of_t = typename result_of< _Tp >::type`
- `using std::true_type = integral_constant< bool, true >`
- `template<typename _Tp >`
`using std::underlying_type_t = typename underlying_type< _Tp >::type`
- `template<typename... >`
`using std::void_t = void`

Functions

- constexpr bool [std::is_constant_evaluated](#) () noexcept
- template<typename _S1, typename _S2, typename _M1, typename _M2 >
constexpr bool [std::is_corresponding_member](#) (_M1 _S1::* __m1, _M2 _S2::* __m2) noexcept
- template<typename _Tp, typename _Mem >
constexpr bool [std::is_pointer_interconvertible_with_class](#) (_Mem _Tp::* __mp) noexcept
- template<typename _Tp >
constexpr _Require<__not_< __is_tuple_like< _Tp > >, [is_move_constructible](#)< _Tp >, [is_move_assignable](#)< _Tp > > [std::swap](#) (_Tp &, _Tp &) noexcept(__and_< [is_nothrow_move_constructible](#)< _Tp >, [is_nothrow_move_assignable](#)< _Tp > >::value)
- template<typename _Tp, size_t _Nm>
constexpr __enable_if_t< __is_swappable< _Tp >::value > [std::swap](#) (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])
noexcept(__is_nothrow_swappable< _Tp >::value)

Variables

- static const size_t [std::aligned_union](#)< _Len, _Types >::alignment_value
- template<typename _From, typename _To >
constexpr bool [std::is_nothrow_convertible_v](#)
- template<typename _Tp >
constexpr bool [std::is_nothrow_swappable_v](#)
- template<typename _Tp, typename _Up >
constexpr bool [std::is_nothrow_swappable_with_v](#)
- template<typename _Tp >
constexpr bool [std::is_swappable_v](#)
- template<typename _Tp, typename _Up >
constexpr bool [std::is_swappable_with_v](#)
- template<typename _Tp >
using [std::remove_cvref_t](#) = typename remove_cvref< _Tp >::type
- #define [__cpp_lib_remove_cvref](#)
- template<typename _Tp >
using [std::type_identity_t](#) = typename type_identity< _Tp >::type
- #define [__cpp_lib_type_identity](#)
- template<typename _Tp >
using [std::unwrap_reference_t](#) = typename [unwrap_reference](#)< _Tp >::type
- template<typename _Tp >
using [std::unwrap_ref_decay_t](#) = typename [unwrap_ref_decay](#)< _Tp >::type
- template<typename... _Bn>
constexpr bool [std::disjunction_v](#)
- template<typename _Pp >
constexpr bool [std::negation_v](#)
- template<typename _Tp >
constexpr bool [std::is_null_pointer_v](#)
- template<typename _Tp >
constexpr bool [std::is_integral_v](#)
- template<typename _Tp >
constexpr bool [std::is_floating_point_v](#)

- `template<typename _Tp >`
`constexpr bool std::is_array_v`
- `template<typename _Tp >`
`constexpr bool std::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_rvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool std::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_const_v`
- `template<typename _Tp >`
`constexpr bool std::is_volatile_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivial_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool std::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::is_empty_v`
- `template<typename _Tp >`
`constexpr bool std::is_polymorphic_v`

- `template<typename _Tp >`
`constexpr bool std::is_abstract_v`
- `template<typename _Tp >`
`constexpr bool std::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::is_unsigned_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_trivially_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_nothrow_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_assignable_v`

- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr size_t std::alignment_of_v`
- `template<typename _Tp >`
`constexpr size_t std::rank_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::extent_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_same_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::is_base_of_v`
- `template<typename _From, typename _To >`
`constexpr bool std::is_convertible_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_r_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_r_v`

3.16.7.1 Detailed Description

Template utilities for compile-time introspection and modification, including type classification traits, type property inspection traits and type transformation traits.

Since

C++11

3.16.7.2 Macro Definition Documentation

`__cpp_lib_remove_cvref`

```
#define __cpp_lib_remove_cvref
```

- Remove references and cv-qualifiers.

Since

C++20

`__cpp_lib_type_identity`

```
#define __cpp_lib_type_identity
```

- Identity metafunction.

Since

C++20

3.16.7.3 Typedef Documentation

add_const_t

```
template<typename _Tp >
using std::add_const_t = typedef typename add_const<_Tp>::type
Alias template for add_const.
```

add_cv_t

```
template<typename _Tp >
using std::add_cv_t = typedef typename add_cv<_Tp>::type
Alias template for add_cv.
```

add_lvalue_reference_t

```
template<typename _Tp >
using std::add_lvalue_reference_t = typedef typename add_lvalue_reference<_Tp>::type
Alias template for add_lvalue_reference.
```

add_pointer_t

```
template<typename _Tp >
using std::add_pointer_t = typedef typename add_pointer<_Tp>::type
Alias template for add_pointer.
```

add_rvalue_reference_t

```
template<typename _Tp >
using std::add_rvalue_reference_t = typedef typename add_rvalue_reference<_Tp>::type
Alias template for add_rvalue_reference.
```

add_volatile_t

```
template<typename _Tp >
using std::add_volatile_t = typedef typename add_volatile<_Tp>::type
Alias template for add_volatile.
```

aligned_storage_t

```
template<size_t _Len, size_t _Align = __alignof__(typename __aligned_storage_msa<_Len>::__type)>
using std::aligned_storage_t = typedef typename aligned_storage<_Len, _Align>::type
Alias template for aligned_storage.
```

bool_constant

```
template<bool __v>
using std::bool_constant = typedef integral_constant<bool, __v>
Alias template for compile-time boolean constant types.
```

Since

C++17

common_type_t

```
template<typename... _Tp>
using std::common_type_t = typedef typename common_type<_Tp...>::type
Alias template for common_type.
```

conditional_t

```
template<bool _Cond, typename _Iftrue , typename _Iffalse >
using std::conditional_t = typedef typename conditional<_Cond, _Iftrue, _Iffalse>::type
Alias template for conditional.
```

decay_t

```
template<typename _Tp >
using std::decay_t = typedef typename decay<_Tp>::type
Alias template for decay.
```

enable_if_t

```
template<bool _Cond, typename _Tp = void>
using std::enable_if_t = typedef typename enable_if<_Cond, _Tp>::type
Alias template for enable_if.
```

false_type

```
using std::false_type = typedef integral_constant<bool, false>
The type used as a compile-time boolean with false value.
```

invoke_result_t

```
template<typename _Fn , typename... _Args>
using std::invoke_result_t = typedef typename invoke_result<_Fn, _Args...>::type
std::invoke_result_t
```

make_signed_t

```
template<typename _Tp >
using std::make_signed_t = typedef typename make_signed<_Tp>::type
Alias template for make_signed.
```

make_unsigned_t

```
template<typename _Tp >
using std::make_unsigned_t = typedef typename make_unsigned<_Tp>::type
Alias template for make_unsigned.
```

remove_all_extents_t

```
template<typename _Tp >
using std::remove_all_extents_t = typedef typename remove_all_extents<_Tp>::type
Alias template for remove_all_extents.
```

remove_const_t

```
template<typename _Tp >
using std::remove_const_t = typedef typename remove_const<_Tp>::type
Alias template for remove_const.
```

remove_cv_t

```
template<typename _Tp >
using std::remove_cv_t = typedef typename remove_cv<_Tp>::type
Alias template for remove_cv.
```

remove_cvref_t

```
template<typename _Tp >
using std::remove_cvref_t = typedef typename remove_cvref<_Tp>::type
```

- Remove references and cv-qualifiers.

Since

C++20

remove_extent_t

```
template<typename _Tp >
using std::remove_extent_t = typedef typename remove_extent<_Tp>::type
Alias template for remove_extent.
```

remove_pointer_t

```
template<typename _Tp >
using std::remove_pointer_t = typedef typename remove_pointer<_Tp>::type
Alias template for remove_pointer.
```

remove_reference_t

```
template<typename _Tp >
using std::remove_reference_t = typedef typename remove_reference<_Tp>::type
Alias template for remove_reference.
```

remove_volatile_t

```
template<typename _Tp >
using std::remove_volatile_t = typedef typename remove_volatile<_Tp>::type
Alias template for remove_volatile.
```

result_of_t

```
template<typename _Tp >
using std::result_of_t = typedef typename result_of<_Tp>::type
Alias template for result_of.
```

true_type

```
using std::true_type = typedef integral_constant<bool, true>
The type used as a compile-time boolean with true value.
```

type_identity_t

```
template<typename _Tp >
using std::type\_identity\_t = typedef typename type_identity<_Tp>::type
```

- Identity metafunction.

Since

C++20

underlying_type_t

```
template<typename _Tp >
using std::underlying\_type\_t = typedef typename underlying\_type<_Tp>::type
Alias template for underlying\_type.
```

void_t

```
template<typename... >
using std::void\_t = typedef void
```

A metafunction that always yields void, used for detecting valid types.

3.16.7.4 Function Documentation**is_constant_evaluated()**

```
constexpr bool std::is\_constant\_evaluated ( ) [inline], [constexpr], [noexcept]
Returns true only when called during constant evaluation.
```

Since

C++20

Referenced by [std::assume_aligned\(\)](#).

is_corresponding_member()

```
template<typename _S1 , typename _S2 , typename _M1 , typename _M2 >
constexpr bool std::is\_corresponding\_member (
    _M1 _S1::* __m1,
    _M2 _S2::* __m2 ) [constexpr], [noexcept]
```

Since

C++20

is_pointer_interconvertible_with_class()

```
template<typename _Tp , typename _Mem >
constexpr bool std::is\_pointer\_interconvertible\_with\_class (
    _Mem _Tp::* __mp ) [constexpr], [noexcept]
```

True if `__mp` points to the first member of a standard-layout type.

Returns

true if `s.*__mp` is pointer-interconvertible with `s`

Since

C++20

swap() [1/2]

```
template<typename _Tp >
constexpr __Require< __not_< __is_tuple_like< _Tp > >, is_move_constructible< _Tp >, is_move_assignable<
_Tp > > std::swap (
    _Tp & __a,
    _Tp & __b ) [inline], [constexpr], [noexcept]
```

Swaps two values.

Parameters

\leftrightarrow _a	A thing of arbitrary type.
\leftrightarrow _b	Another thing of arbitrary type.

Returns

Nothing.

swap() [2/2]

```
template<typename _Tp , size_t _Nm>
constexpr __enable_if_t< __is_swappable< _Tp >::value > std::swap (
    _Tp (&) __a[_Nm],
    _Tp (&) __b[_Nm] ) [inline], [constexpr], [noexcept]
```

Swap the contents of two arrays.

References [std::swap\(\)](#).

3.16.7.5 Variable Documentation**alignment_value**

```
template<size_t _Len, typename... _Types>
const size_t std::aligned_union< _Len, _Types >::alignment_value [static]
The value of the strictest alignment of _Types.
```

is_nothrow_convertible_v

```
template<typename _From , typename _To >
constexpr bool std::is_nothrow_convertible_v [inline], [constexpr]
is_nothrow_convertible_v
```

is_nothrow_swappable_v

```
template<typename _Tp >
constexpr bool std::is_nothrow_swappable_v [inline], [constexpr]
is_nothrow_swappable_v
```

is_nothrow_swappable_with_v

```
template<typename _Tp , typename _Up >
constexpr bool std::is_nothrow_swappable_with_v [inline], [constexpr]
is_nothrow_swappable_with_v
```

is_swappable_v

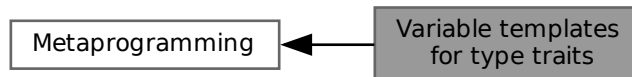
```
template<typename _Tp >
constexpr bool std::is_swappable_v [inline], [constexpr]
is_swappable_v
```

is_swappable_with_v

```
template<typename _Tp , typename _Up >
constexpr bool std::is_swappable_with_v [inline], [constexpr]
is_swappable_with_v
```

3.16.7.6 Variable templates for type traits

Collaboration diagram for Variable templates for type traits:

**Variables**

- template<typename _Tp >
constexpr bool **std::has_unique_object_representations_v**
- template<typename _Tp >
constexpr bool **std::is_aggregate_v**
- template<typename _Tp >
constexpr bool **std::is_bounded_array_v**
- template<typename _Tp , typename _Up >
constexpr bool **std::is_layout_compatible_v**
- template<typename _Base , typename _Derived >
constexpr bool **std::is_pointer_interconvertible_base_of_v**
- template<typename _Tp >
constexpr bool **std::is_unbounded_array_v**

- template<typename... _Bn>
constexpr bool **std::conjunction_v**
- template<typename... _Bn>
constexpr bool **std::disjunction_v**
- template<typename _Pp >
constexpr bool **std::negation_v**

- template<typename _Tp >
constexpr bool **std::is_void_v**
- template<typename _Tp >
constexpr bool **std::is_null_pointer_v**
- template<typename _Tp >
constexpr bool **std::is_integral_v**

- `template<typename _Tp >`
`constexpr bool std::is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool std::is_array_v`
- `template<typename _Tp >`
`constexpr bool std::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_rvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool std::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_const_v`
- `template<typename _Tp >`
`constexpr bool std::is_volatile_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivial_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool std::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::is_empty_v`

- `template<typename _Tp >`
`constexpr bool std::is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool std::is_abstract_v`
- `template<typename _Tp >`
`constexpr bool std::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::is_unsigned_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_trivially_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_nothrow_assignable_v`

- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr size_t std::alignment_of_v`
- `template<typename _Tp >`
`constexpr size_t std::rank_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::extent_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_same_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::is_base_of_v`
- `template<typename _From, typename _To >`
`constexpr bool std::is_convertible_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_r_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_r_v`

3.16.7.6.1 Detailed Description

Each variable `is_xxx_v<T>` is a boolean constant with the same value as the `value` member of the corresponding type trait `is_xxx<T>`.

Since

C++17 unless noted otherwise.

3.16.7.6.2 Variable Documentation

is_bounded_array_v

```
template<typename _Tp >
constexpr bool std::is_bounded_array_v [inline], [constexpr]
```

Since

C++20

is_layout_compatible_v

```
template<typename _Tp, typename _Up >
constexpr bool std::is_layout_compatible_v [constexpr]
```

Since

C++20

is_pointer_interconvertible_base_of_v

```
template<typename _Base , typename _Derived >
constexpr bool std::is_pointer_interconvertible_base_of_v [constexpr]
```

Since

C++20

is_unbounded_array_v

```
template<typename _Tp >
constexpr bool std::is_unbounded_array_v [inline], [constexpr]
```

Since

C++20

3.16.8 Rational Arithmetic

Collaboration diagram for Rational Arithmetic:

**Files**

- file [ratio](#)

Classes

- struct [std::ratio](#)< _Num, _Den >
- struct [std::ratio_equal](#)< _R1, _R2 >
- struct [std::ratio_greater](#)< _R1, _R2 >
- struct [std::ratio_greater_equal](#)< _R1, _R2 >
- struct [std::ratio_less](#)< _R1, _R2 >
- struct [std::ratio_less_equal](#)< _R1, _R2 >
- struct [std::ratio_not_equal](#)< _R1, _R2 >

Typedefs

- typedef [ratio](#)< 1, 10000000000000000000 > **std::atto**
- typedef [ratio](#)< 1, 100 > **std::centi**
- typedef [ratio](#)< 10, 1 > **std::deca**
- typedef [ratio](#)< 1, 10 > **std::deci**
- typedef [ratio](#)< 1000000000000000000, 1 > **std::exa**
- typedef [ratio](#)< 1, 1000000000000000 > **std::femto**
- typedef [ratio](#)< 1000000000, 1 > **std::giga**

- typedef `ratio< 100, 1 > std::hecto`
- typedef `ratio< 1000, 1 > std::kilo`
- typedef `ratio< 1000000, 1 > std::mega`
- typedef `ratio< 1, 1000000 > std::micro`
- typedef `ratio< 1, 1000 > std::milli`
- typedef `ratio< 1, 1000000000 > std::nano`
- typedef `ratio< 1000000000000000, 1 > std::peta`
- typedef `ratio< 1, 1000000000000 > std::pico`
- template<typename _R1, typename _R2 >
using `std::ratio_add` = typename `__ratio_add< _R1, _R2 >::type`
- template<typename _R1, typename _R2 >
using `std::ratio_divide` = typename `__ratio_divide< _R1, _R2 >::type`
- template<typename _R1, typename _R2 >
using `std::ratio_multiply` = typename `__ratio_multiply< _R1, _R2 >::type`
- template<typename _R1, typename _R2 >
using `std::ratio_subtract` = typename `__ratio_subtract< _R1, _R2 >::type`
- typedef `ratio< 1000000000000, 1 > std::tera`

Variables

- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_equal_v`
- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_greater_equal_v`
- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_greater_v`
- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_less_equal_v`
- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_less_v`
- template<typename _R1, typename _R2 >
constexpr bool `std::ratio_not_equal_v`

3.16.8.1 Detailed Description

Compile time representation of finite rational numbers.

3.16.8.2 Typedef Documentation

`ratio_add`

```
template<typename _R1, typename _R2 >
using std::ratio_add = typedef typename __ratio_add<_R1, _R2>::type
ratio_add
```

`ratio_divide`

```
template<typename _R1, typename _R2 >
using std::ratio_divide = typedef typename __ratio_divide<_R1, _R2>::type
ratio_divide
```

ratio_multiply

```
template<typename _R1 , typename _R2 >
using std::ratio\_multiply = typedef typename __ratio_multiply<_R1, _R2>::type
ratio_multiply
```

ratio_subtract

```
template<typename _R1 , typename _R2 >
using std::ratio\_subtract = typedef typename __ratio_subtract<_R1, _R2>::type
ratio_subtract
```

3.16.9 Time

Collaboration diagram for Time:

**Files**

- file [chrono](#)

Namespaces

- namespace [std::chrono](#)
- namespace [std::literals::chrono_literals](#)

Classes

- struct [std::common_type< chrono::duration< _Rep, _Period > >](#)
- struct [std::common_type< chrono::duration< _Rep, _Period >, chrono::duration< _Rep, _Period > >](#)
- struct [std::common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >](#)
- class [std::chrono::duration< _Rep, _Period >](#)
- struct [std::chrono::duration_values< _Rep >](#)
- struct [std::chrono::_V2::steady_clock](#)
- struct [std::chrono::_V2::system_clock](#)
- class [std::chrono::time_point< _Clock, _Dur >](#)
- struct [std::chrono::treat_as_floating_point< _Rep >](#)

Macros

- [#define __cpp_lib_chrono](#)

Typedefs

- using `std::chrono::days` = `duration< int64_t, ratio< 86400 > >`
- using `std::chrono::file_clock` = `::std::filesystem::__file_clock`
- template<typename `_Duration` >
using `std::chrono::file_time` = `time_point< file_clock, _Duration >`
- using `std::chrono::gps_seconds` = `gps_time< seconds >`
- template<typename `_Duration` >
using `std::chrono::gps_time` = `time_point< gps_clock, _Duration >`
- using `std::chrono::high_resolution_clock` = `system_clock`
- using `std::chrono::hours` = `duration< int64_t, ratio< 3600 > >`
- using `std::chrono::local_days` = `local_time< days >`
- using `std::chrono::local_seconds` = `local_time< seconds >`
- template<typename `_Duration` >
using `std::chrono::local_time` = `time_point< local_t, _Duration >`
- using `std::chrono::microseconds` = `duration< int64_t, micro >`
- using `std::chrono::milliseconds` = `duration< int64_t, milli >`
- using `std::chrono::minutes` = `duration< int64_t, ratio< 60 > >`
- using `std::chrono::months` = `duration< int64_t, ratio< 2629746 > >`
- using `std::chrono::nanoseconds` = `duration< int64_t, nano >`
- using `std::chrono::seconds` = `duration< int64_t >`
- using `std::chrono::sys_days` = `sys_time< days >`
- using `std::chrono::sys_seconds` = `sys_time< seconds >`
- template<typename `_Duration` >
using `std::chrono::sys_time` = `time_point< system_clock, _Duration >`
- using `std::chrono::tai_seconds` = `tai_time< seconds >`
- template<typename `_Duration` >
using `std::chrono::tai_time` = `time_point< tai_clock, _Duration >`
- using `std::chrono::utc_seconds` = `utc_time< seconds >`
- template<typename `_Duration` >
using `std::chrono::utc_time` = `time_point< utc_clock, _Duration >`
- using `std::chrono::weeks` = `duration< int64_t, ratio< 604800 > >`
- using `std::chrono::years` = `duration< int64_t, ratio< 31556952 > >`

Functions

- constexpr `std::chrono::year_month_day::year_month_day` (const `year_month_day_last` & `__ymdl`) noexcept
- template<typename `_Rep`, typename `_Period` >
constexpr `enable_if_t< numeric_limits< _Rep >::is_signed, duration< _Rep, _Period > >` `std::chrono::abs` (`duration< _Rep, _Period > __d`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration< _ToDur >` `std::chrono::ceil` (const `duration< _Rep, _Period > &__d`)
- template<typename `_ToDur`, typename `_Clock`, typename `_Dur` >
constexpr `enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > >` `std::chrono::ceil` (const `time_point< _Clock, _Dur > &__tp`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration< _ToDur >` `std::chrono::duration_cast` (const `duration< _Rep, _Period > &__d`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration< _ToDur >` `std::chrono::floor` (const `duration< _Rep, _Period > &__d`)
- template<typename `_ToDur`, typename `_Clock`, typename `_Dur` >
constexpr `enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > >` `std::chrono::floor` (const `time_point< _Clock, _Dur > &__tp`)

- constexpr bool **std::chrono::is_am** (const hours &__h) noexcept
- constexpr bool **std::chrono::is_pm** (const hours &__h) noexcept
- constexpr hours **std::chrono::make12** (const hours &__h) noexcept
- constexpr hours **std::chrono::make24** (const hours &__h, bool __is_pm) noexcept
- constexpr bool **std::chrono::year_month_day::ok** () const noexcept
- constexpr chrono::day **std::literals::chrono_literals::operator""d** (unsigned long long __d) noexcept
- template<char... _Digits>
constexpr chrono::hours **std::literals::chrono_literals::operator""h** ()
- constexpr chrono::duration< long double, ratio< 3600, 1 > > **std::literals::chrono_literals::operator""h** (long double __hours)
- template<char... _Digits>
constexpr chrono::minutes **std::literals::chrono_literals::operator""min** ()
- constexpr chrono::duration< long double, ratio< 60, 1 > > **std::literals::chrono_literals::operator""min** (long double __mins)
- template<char... _Digits>
constexpr chrono::milliseconds **std::literals::chrono_literals::operator""ms** ()
- constexpr chrono::duration< long double, milli > **std::literals::chrono_literals::operator""ms** (long double __msecs)
- template<char... _Digits>
constexpr chrono::nanoseconds **std::literals::chrono_literals::operator""ns** ()
- constexpr chrono::duration< long double, nano > **std::literals::chrono_literals::operator""ns** (long double __nsecs)
- template<char... _Digits>
constexpr chrono::seconds **std::literals::chrono_literals::operator""s** ()
- constexpr chrono::duration< long double > **std::literals::chrono_literals::operator""s** (long double __secs)
- template<char... _Digits>
constexpr chrono::microseconds **std::literals::chrono_literals::operator""us** ()
- constexpr chrono::duration< long double, micro > **std::literals::chrono_literals::operator""us** (long double __usecs)
- constexpr chrono::year **std::literals::chrono_literals::operator""y** (unsigned long long __y) noexcept
- template<typename _CharT, typename _Traits, typename _Rep, typename _Period >
basic_ostream< _CharT, _Traits > & **std::chrono::operator**<< (**std::basic_ostream**< _CharT, _Traits > &__os, const duration< _Rep, _Period > &__d)
- constexpr weekday_last **std::chrono::weekday::operator[]** (last_spec) const noexcept
- constexpr weekday_indexed **std::chrono::weekday::operator[]** (unsigned __index) const noexcept
- template<typename _ToDur, typename _Rep, typename _Period >
constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename _ToDur::rep > > >::value, _ToDur > **std::chrono::round** (const duration< _Rep, _Period > &__d)
- template<typename _ToDur, typename _Clock, typename _Dur >
constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename _ToDur::rep > > >::value, time_point< _Clock, _ToDur > > **std::chrono::round** (const time_point< _Clock, _Dur > &__tp)
- template<typename _ToDur, typename _Clock, typename _Dur >
constexpr __enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > **std::chrono::time_point_cast** (const time_point< _Clock, _Dur > &__t)

Variables

- constexpr month **std::chrono::April**
- constexpr month **std::chrono::August**
- constexpr month **std::chrono::December**
- constexpr month **std::chrono::February**
- constexpr weekday **std::chrono::Friday**
- template<typename _Tp >
constexpr bool **std::chrono::is_clock_v**

- `template<> constexpr bool std::chrono::is_clock_v< file_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< gps_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< steady_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< system_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< tai_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< utc_clock >`
- `constexpr month std::chrono::January`
- `constexpr month std::chrono::July`
- `constexpr month std::chrono::June`
- `constexpr last_spec std::chrono::last`
- `constexpr month std::chrono::March`
- `constexpr month std::chrono::May`
- `constexpr weekday std::chrono::Monday`
- `constexpr month std::chrono::November`
- `constexpr month std::chrono::October`
- `constexpr weekday std::chrono::Saturday`
- `constexpr month std::chrono::September`
- `constexpr weekday std::chrono::Sunday`
- `constexpr weekday std::chrono::Thursday`
- `template<typename _Rep >`
`constexpr bool std::chrono::treat_as_floating_point_v`
- `constexpr weekday std::chrono::Tuesday`
- `constexpr weekday std::chrono::Wednesday`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::chrono::operator-`
`(const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type operator+`
`(const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Rep2, typename _Period >`
`constexpr duration< __common_rep_t< _Rep2, _Rep1 >, _Period > std::chrono::operator* (const _Rep1 &__s,`
`const duration< _Rep2, _Period > &__d)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period >`
`std::chrono::operator/ (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< _Rep1, _Rep2 >::type std::chrono::operator/ (const duration< _Rep1, _Period1 >`
`&__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period >`
`std::chrono::operator% (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::chrono::operator%`
`(const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, _Rep2 >, _Period > operator* (const duration< _Rep1, _`
`Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator< (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2,`
`_Period2 > &__rhs)`

- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator!= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2,`
`_Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator<= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2,`
`_Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator> (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2,`
`_Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator>= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2,`
`_Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator== (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 >`
`&__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Clock, typename _Dur2 >`
`constexpr time_point< _Clock, typename common_type< duration< _Rep1, _Period1 >, _Dur2 >::type >`
`std::chrono::operator+ (const duration< _Rep1, _Period1 > &__lhs, const time_point< _Clock, _Dur2 > &__`
`__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >`
`constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type >`
`std::chrono::operator- (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__`
`__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr common_type< _Dur1, _Dur2 >::type std::chrono::operator- (const time_point< _Clock, _Dur1 > &__`
`__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >`
`constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type >`
`operator+ (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator!= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock,`
`_Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator< (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock,`
`_Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator<= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _`
`_Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator> (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock,`
`_Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator>= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _`
`_Clock, _Dur2 > &__rhs)`

3.16.9.1 Detailed Description

Classes and functions for time.

Since

C++11

3.16.9.2 Typedef Documentation

days

```
using std::chrono::days = typedef duration<int64_t, ratio<86400> >  
days
```

high_resolution_clock

```
using std::chrono::_V2::high_resolution_clock = typedef system_clock  
Highest-resolution clock.
```

This is the clock "with the shortest tick period." Alias to `std::system_clock` until higher-than-nanosecond definitions become feasible.

hours

```
using std::chrono::hours = typedef duration<int64_t, ratio<3600> >  
hours
```

microseconds

```
using std::chrono::microseconds = typedef duration<int64_t, micro>  
microseconds
```

milliseconds

```
using std::chrono::milliseconds = typedef duration<int64_t, milli>  
milliseconds
```

minutes

```
using std::chrono::minutes = typedef duration<int64_t, ratio< 60> >  
minutes
```

months

```
using std::chrono::months = typedef duration<int64_t, ratio<2629746> >  
months
```

nanoseconds

```
using std::chrono::nanoseconds = typedef duration<int64_t, nano>  
nanoseconds
```

seconds

```
using std::chrono::seconds = typedef duration<int64_t>  
seconds
```

weeks

```
using std::chrono::weeks = typedef duration<int64_t, ratio<604800> >  
weeks
```

years

```
using std::chrono::years = typedef duration<int64_t, ratio<31556952> >  
years
```

3.16.9.3 Function Documentation

abs()

```
template<typename _Rep , typename _Period >  
constexpr enable_if_t< numeric_limits< _Rep >::is_signed, duration< _Rep, _Period > > std::  
::chrono::abs (   
    duration< _Rep, _Period > __d ) [constexpr]
```

The absolute (non-negative) value of a duration.

Parameters

$_d$	A duration with a signed rep type.
-------	------------------------------------

Returns

A duration of the same type as the argument, with value $|d|$.

Since

C++17

References [std::abs\(\)](#).

ceil() [1/2]

```
template<typename _ToDur , typename _Rep , typename _Period >  
constexpr __enable_if_is_duration< _ToDur > std::chrono::ceil (   
    const duration< _Rep, _Period > & __d ) [constexpr]
```

Convert a duration to type `ToDur` and round up.

If the duration cannot be represented exactly in the result type, returns the closest value that is greater than the argument.

Template Parameters

$_ToDur$	The result type must be a duration.
-----------	-------------------------------------

Parameters

$_d$	A duration.
-------	-------------

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++17

ceil() [2/2]

```
template<typename _ToDur , typename _Clock , typename _Dur >
constexpr enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > std::
::chrono::ceil (
    const time_point< _Clock, _Dur > & __tp ) [constexpr]
```

Convert a `time_point` to type `ToDur` and round up.

The result is the same time point as measured by the same clock, but using the specified `duration` to represent the time. If the time point cannot be represented exactly in the result type, returns the closest value that is greater than the argument.

Template Parameters

<code>_ToDur</code>	The duration type to use for the result.
---------------------	--

Parameters

<code>__tp</code>	A time point.
-------------------	---------------

Returns

The value of `__tp` converted to type `_ToDur`.

Since

C++17

duration_cast()

```
template<typename _ToDur , typename _Rep , typename _Period >
constexpr __enable_if_is_duration< _ToDur > std::chrono::duration_cast (
    const duration< _Rep, _Period > & __d ) [constexpr]
```

Convert a `duration` to type `ToDur`.

If the duration cannot be represented accurately in the result type, returns the result of integer truncation (i.e., rounded towards zero).

Template Parameters

<code>_ToDur</code>	The result type must be a duration.
---------------------	-------------------------------------

Parameters

<code>__d</code>	A duration.
------------------	-------------

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++11

floor() [1/2]

```
template<typename _ToDur , typename _Rep , typename _Period >
constexpr __enable_if_is_duration< _ToDur > std::chrono::floor (
    const duration< _Rep, _Period > & __d ) [constexpr]
```

Convert a duration to type `ToDur` and round down.

If the duration cannot be represented exactly in the result type, returns the closest value that is less than the argument.

Template Parameters

<code>_ToDur</code>	The result type must be a duration.
---------------------	-------------------------------------

Parameters

<code>__↔ _d</code>	A duration.
-------------------------	-------------

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++17

floor() [2/2]

```
template<typename _ToDur , typename _Clock , typename _Dur >
constexpr enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > std↔
::chrono::floor (
    const time_point< _Clock, _Dur > & __tp ) [constexpr]
```

Convert a `time_point` to type `ToDur` and round down.

The result is the same time point as measured by the same clock, but using the specified `duration` to represent the time. If the time point cannot be represented exactly in the result type, returns the closest value that is less than the argument.

Template Parameters

<code>_ToDur</code>	The <code>duration</code> type to use for the result.
---------------------	---

Parameters

↵	A time point.
_↵	
↵	
_↵	
<i>t</i>	

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++17

operator"!=()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool std::chrono::operator!= (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Comparisons for `chrono::duration`

operator""d()

```
constexpr chrono::day std::literals::chrono_literals::operator""d (
    unsigned long long __d ) [constexpr], [noexcept]
```

Literal suffix for creating `chrono::day` objects.

Since

C++20

operator""h() [1/2]

```
template<char... _Digits>
constexpr chrono::hours std::literals::chrono_literals::operator""h ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::hours`

operator""h() [2/2]

```
constexpr chrono::duration< long double, ratio< 3600, 1 > > std::literals::chrono_literals::operator""h (
    long double __hours ) [constexpr]
```

Literal suffix for durations representing non-integer hours.

operator""min() [1/2]

```
template<char... _Digits>
constexpr chrono::minutes std::literals::chrono_literals::operator""min ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::minutes`

operator""min() [2/2]

```
constexpr chrono::duration< long double, ratio< 60, 1 > > std::literals::chrono_literals::operator""min (
    long double __mins ) [constexpr]
```

Literal suffix for durations representing non-integer minutes.

operator""ms() [1/2]

```
template<char... _Digits>
constexpr chrono::milliseconds std::literals::chrono_literals::operator""ms ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::milliseconds`

operator""ms() [2/2]

```
constexpr chrono::duration< long double, milli > std::literals::chrono_literals::operator""ms (
    long double __msecs ) [constexpr]
```

Literal suffix for durations representing non-integer milliseconds.

operator""ns() [1/2]

```
template<char... _Digits>
constexpr chrono::nanoseconds std::literals::chrono_literals::operator""ns ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::nanoseconds`

operator""ns() [2/2]

```
constexpr chrono::duration< long double, nano > std::literals::chrono_literals::operator""ns (
    long double __nsecs ) [constexpr]
```

Literal suffix for durations representing non-integer nanoseconds.

operator""s() [1/2]

```
template<char... _Digits>
constexpr chrono::seconds std::literals::chrono_literals::operator""s ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::seconds`

operator""s() [2/2]

```
constexpr chrono::duration< long double > std::literals::chrono_literals::operator""s (
    long double __secs ) [constexpr]
```

Literal suffix for durations representing non-integer seconds.

operator""us() [1/2]

```
template<char... _Digits>
constexpr chrono::microseconds std::literals::chrono_literals::operator""us ( ) [constexpr]
```

Literal suffix for durations of type `std::chrono::microseconds`

operator""us() [2/2]

```
constexpr chrono::duration< long double, micro > std::literals::chrono_literals::operator""us (
    long double __usecs ) [constexpr]
```

Literal suffix for durations representing non-integer microseconds.

operator""y()

```
constexpr chrono::year std::literals::chrono_literals::operator""y (
    unsigned long long __y ) [constexpr], [noexcept]
```

Literal suffix for creating chrono::year objects.

Since

C++20

operator%() [1/2]

```
template<typename _Rep1 , typename _Period , typename _Rep2 >
constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > std::
::chrono::operator% (
    const duration< _Rep1, _Period > & __d,
    const _Rep2 & __s ) [constexpr]
```

Arithmetic operators for chrono::duration

operator%() [2/2]

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::
::chrono::operator% (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Arithmetic operators for chrono::duration

operator*() [1/2]

```
template<typename _Rep1 , typename _Rep2 , typename _Period >
constexpr duration< __common_rep_t< _Rep2, _Rep1 >, _Period > std::chrono::operator* (
    const _Rep1 & __s,
    const duration< _Rep2, _Period > & __d ) [constexpr]
```

Arithmetic operators for chrono::duration

operator*() [2/2]

```
template<typename _Rep1 , typename _Period , typename _Rep2 >
constexpr duration< __common_rep_t< _Rep1, _Rep2 >, _Period > operator* (
    const duration< _Rep1, _Period > & __d,
    const _Rep2 & __s ) [related]
```

Arithmetic operators for chrono::duration

operator+() [1/3]

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type operator+
(
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [related]
```

The sum of two durations.

operator+() [2/3]

```
template<typename _Rep1 , typename _Period1 , typename _Clock , typename _Dur2 >
constexpr time_point< _Clock, typename common_type< duration< _Rep1, _Period1 >, _Dur2 >::type >
std::chrono::operator+ (
    const duration< _Rep1, _Period1 > & __lhs,
    const time_point< _Clock, _Dur2 > & __rhs ) [constexpr]
```

Adjust a time point forwards by the given duration.

operator+() [3/3]

```
template<typename _Clock , typename _Dur1 , typename _Rep2 , typename _Period2 >
constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type >
operator+ (
    const time_point< _Clock, _Dur1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [related]
```

Adjust a time point forwards by the given duration.

operator-() [1/3]

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::
::chrono::operator- (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

The difference between two durations.

operator-() [2/3]

```
template<typename _Clock , typename _Dur1 , typename _Rep2 , typename _Period2 >
constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type >
std::chrono::operator- (
    const time_point< _Clock, _Dur1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Adjust a time point backwards by the given duration.

operator-() [3/3]

```
template<typename _Clock , typename _Dur1 , typename _Dur2 >
constexpr common_type< _Dur1, _Dur2 >::type std::chrono::operator- (
    const time_point< _Clock, _Dur1 > & __lhs,
    const time_point< _Clock, _Dur2 > & __rhs ) [constexpr]
```

The difference between two time points (as a duration)

operator/() [1/2]

```
template<typename _Rep1 , typename _Period , typename _Rep2 >
constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > std::
::chrono::operator/ (
    const duration< _Rep1, _Period > & __d,
    const _Rep2 & __s ) [constexpr]
```

Arithmetic operators for chrono::duration

operator/() [2/2]

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< _Rep1, _Rep2 >::type std::chrono::operator/ (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Arithmetic operators for chrono::duration

operator<()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool std::chrono::operator< (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Comparisons for chrono::duration

operator<=()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool std::chrono::operator<= (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Comparisons for chrono::duration

operator==()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool operator== (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [related]
```

Comparisons for chrono::duration

operator>()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool std::chrono::operator> (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Comparisons for chrono::duration

operator>=()

```
template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool std::chrono::operator>= (
    const duration< _Rep1, _Period1 > & __lhs,
    const duration< _Rep2, _Period2 > & __rhs ) [constexpr]
```

Comparisons for chrono::duration

round() [1/2]

```
template<typename _ToDur , typename _Rep , typename _Period >
constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename
_ToDur::rep > > >::value, _ToDur > std::chrono::round (
    const duration< _Rep, _Period > & __d ) [constexpr]
```

Convert a duration to type ToDur and round to the closest value.

If the duration cannot be represented exactly in the result type, returns the closest value, rounding ties to even.

Template Parameters

<code>_ToDur</code>	The result type must be a <code>duration</code> with a non-floating-point <code>rep</code> type.
---------------------	--

Parameters

<code>↔</code> <code>_d</code>	A duration.
-----------------------------------	-------------

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++17

round() [2/2]

```
template<typename _ToDur , typename _Clock , typename _Dur >
constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename
_ToDur::rep > > >::value, time_point< _Clock, _ToDur > > std::chrono::round (
    const time_point< _Clock, _Dur > & __tp ) [constexpr]
```

Convert a `time_point` to type `ToDur` and round to the closest value.

The result is the same time point as measured by the same clock, but using the specified `duration` to represent the time. If the time point cannot be represented exactly in the result type, returns the closest value, rounding ties to even.

Template Parameters

<code>_ToDur</code>	The <code>duration</code> type to use for the result, which must have a non-floating-point <code>rep</code> type.
---------------------	---

Parameters

<code>↔</code> <code>↔</code> <code>↔</code> <code>↔</code> <code>t</code>	A time point.
--	---------------

Returns

The value of `__d` converted to type `_ToDur`.

Since

C++17

time_point_cast()

```
template<typename _ToDur , typename _Clock , typename _Dur >
constexpr __enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > std::
```

```

::chrono::time_point_cast (
    const time\_point< _Clock, _Dur > & __t ) [constexpr]

```

Convert a `time_point` to use duration type `ToDur`.

The result is the same time point as measured by the same clock, but using the specified `duration` to represent the time. If the time point cannot be represented accurately in the result type, returns the result of integer truncation (i.e., rounded towards zero).

Template Parameters

<code>_ToDur</code>	The <code>duration</code> type to use for the result.
---------------------	---

Parameters

<code>↩</code>	A time point.
<code>↩</code>	
<code>↩</code>	
<code>↩</code>	
<code>t</code>	

Returns

The value of `__t` converted to use type `_ToDur`.

Since

C++11

4 Namespace Documentation

4.1 `__gnu_cxx` Namespace Reference

Namespaces

- namespace [__detail](#)
- namespace [typelist](#)

Classes

- struct [__alloc_traits](#)
- struct [__common_pool_policy](#)
- class [__mt_alloc](#)
- class [__mt_alloc_base](#)
- struct [__per_type_pool_policy](#)
- class [__pool](#)
- class [__pool< false >](#)
- class [__pool< true >](#)
- class [__pool_alloc](#)
- class [__pool_alloc_base](#)
- struct [__pool_base](#)
- class [__rc_string_base](#)
- class [__scoped_lock](#)
- class [__versa_string](#)

- struct `_Caster`
- struct `_Char_types`
- class `_ExtPtr_allocator`
- struct `_Invalid_type`
- class `_Pointer_adapter`
- class `_Relative_pointer_impl`
- class `_Relative_pointer_impl< const _Tp >`
- class `_Std_pointer_impl`
- struct `_Unqualified_type`
- struct `annotate_base`
- class `binary_compose`
- class `bitmap_allocator`
- struct `char_traits`
- struct `character`
- struct `condition_base`
- struct `constant_binary_fun`
- struct `constant_unary_fun`
- struct `constant_void_fun`
- class `debug_allocator`
- class `enc_filebuf`
- struct `encoding_char_traits`
- class `encoding_state`
- struct `forced_error`
- class `free_list`
- class `hash_map`
- class `hash_multimap`
- class `hash_multiset`
- class `hash_set`
- struct `limit_condition`
- class `malloc_allocator`
- class `new_allocator`
- struct `project1st`
- struct `project2nd`
- struct `random_condition`
- struct `rb_tree`
- class `recursive_init_error`
- class `rope`
- struct `select1st`
- struct `select2nd`
- class `slist`
- class `stdio_filebuf`
- class `stdio_sync_filebuf`
- class `subtractive_rng`
- struct `temporary_buffer`
- class `throw_allocator_base`
- struct `throw_allocator_limit`
- struct `throw_allocator_random`
- struct `throw_value_base`
- struct `throw_value_limit`
- struct `throw_value_random`
- class `unary_compose`

Typedefs

- typedef void(* **__destroy_handler**) (void *)
- template<typename _Tp >
using **__int_traits** = __numeric_traits_integer< _Tp >
- typedef **__versa_string**< char, [std::char_traits](#)< char >, [std::allocator](#)< char >, [__rc_string_base](#) > **__rc_string**
- typedef **__vstring** **__sso_string**
- typedef **__versa_string**< char16_t, [std::char_traits](#)< char16_t >, [std::allocator](#)< char16_t >, [__rc_string_base](#) > **__u16rc_string**
- typedef **__u16vstring** **__u16sso_string**
- typedef **__versa_string**< char16_t > **__u16vstring**
- typedef **__versa_string**< char32_t, [std::char_traits](#)< char32_t >, [std::allocator](#)< char32_t >, [__rc_string_base](#) > **__u32rc_string**
- typedef **__u32vstring** **__u32sso_string**
- typedef **__versa_string**< char32_t > **__u32vstring**
- typedef **__versa_string**< char > **__vstring**
- typedef **__versa_string**< wchar_t, [std::char_traits](#)< wchar_t >, [std::allocator](#)< wchar_t >, [__rc_string_base](#) > **__wrc_string**
- typedef **__wvstring** **__wsso_string**
- typedef **__versa_string**< wchar_t > **__wvstring**
- typedef [rope](#)< char > **crope**
- typedef [rope](#)< wchar_t > **wrope**

Enumerations

- enum { **_S_num_primes** }
- enum **_Lock_policy** { **_S_single** , **_S_mutex** , **_S_atomic** }

Functions

- void **__atomic_add** (volatile _Atomic_word *, int) noexcept
- void **__atomic_add_dispatch** (_Atomic_word * __mem, int __val)
- void **__atomic_add_single** (_Atomic_word * __mem, int __val)
- template<class _Tp >
void **__aux_require_boolean_expr** (const _Tp & __t)
- template<typename _ToType , typename _FromType >
__ToType **__const_pointer_cast** (_FromType * __arg)
- template<typename _ToType , typename _FromType >
__ToType **__const_pointer_cast** (const _FromType & __arg)
- template<typename _InputIterator , typename _Size , typename _OutputIterator >
[std::pair](#)< _InputIterator, _OutputIterator > **__copy_n** (_InputIterator __first, _Size __count, _OutputIterator __↔
__result, [std::input_iterator_tag](#))
- template<typename _RAIterator , typename _Size , typename _OutputIterator >
[std::pair](#)< _RAIterator, _OutputIterator > **__copy_n** (_RAIterator __first, _Size __count, _OutputIterator __result,
[std::random_access_iterator_tag](#))
- template<typename _InputIterator , typename _Distance >
void **__distance** (_InputIterator __first, _InputIterator __last, _Distance & __n, [std::input_iterator_tag](#))
- template<typename _RandomAccessIterator , typename _Distance >
void **__distance** (_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance & __n, [std::random_access_iterator_tag](#))
- template<typename _ToType , typename _FromType >
__ToType **__dynamic_pointer_cast** (_FromType * __arg)
- template<typename _ToType , typename _FromType >
__ToType **__dynamic_pointer_cast** (const _FromType & __arg)

- `void __error_type_must_be_a_signed_integer_type ()`
- `void __error_type_must_be_an_integer_type ()`
- `void __error_type_must_be_an_unsigned_integer_type ()`
- `_Atomic_word __exchange_and_add (volatile _Atomic_word *, int) noexcept`
- `_Atomic_word __exchange_and_add_dispatch (_Atomic_word *__mem, int __val)`
- `_Atomic_word __exchange_and_add_single (_Atomic_word *__mem, int __val)`
- `template<class _Concept >`
`constexpr void __function_requires ()`
- `template<typename _Type >`
`constexpr bool __is_null_pointer (_Type *__ptr)`
- `template<typename _Type >`
`constexpr bool __is_null_pointer (_Type)`
- `constexpr bool __is_null_pointer (std::nullptr_t)`
- `bool __is_single_threaded () noexcept`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int __lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `int __lexicographical_compare_3way (const char *__first1, const char *__last1, const char *__first2, const char *__last2)`
- `int __lexicographical_compare_3way (const unsigned char *__first1, const unsigned char *__last1, const unsigned char *__first2, const unsigned char *__last2)`
- `template<typename _Tp >`
`const _Tp & __median (const _Tp &__a, const _Tp &__b, const _Tp &__c)`
- `template<typename _Tp, typename _Compare >`
`const _Tp & __median (const _Tp &__a, const _Tp &__b, const _Tp &__c, _Compare __comp)`
- `crope::reference __mutable_reference_at (crope &__c, std::size_t __i)`
- `template<typename _Tp, typename _Integer >`
`_Tp __power (_Tp __x, _Integer __n)`
- `template<typename _Tp, typename _Integer, typename _MonoidOperation >`
`_Tp __power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator, typename _Distance >`
`_RandomAccessIterator __random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, _RandomNumberGenerator &__rand, const _Distance __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Distance >`
`_RandomAccessIterator __random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, const _Distance __n)`
- `template<typename _ToType, typename _FromType >`
`_ToType __reinterpret_pointer_cast (_FromType *__arg)`
- `template<typename _ToType, typename _FromType >`
`_ToType __reinterpret_pointer_cast (const _FromType &__arg)`
- `_Slist_node_base * __slist_make_link (_Slist_node_base *__prev_node, _Slist_node_base *__new_node)`
- `_Slist_node_base * __slist_previous (_Slist_node_base *__head, const _Slist_node_base *__node)`
- `const _Slist_node_base * __slist_previous (const _Slist_node_base *__head, const _Slist_node_base *__node)`
- `_Slist_node_base * __slist_reverse (_Slist_node_base *__node)`
- `std::size_t __slist_size (_Slist_node_base *__node)`
- `void __slist_splice_after (_Slist_node_base *__pos, _Slist_node_base *__before_first, _Slist_node_base *__before_last)`
- `void __slist_splice_after (_Slist_node_base *__pos, _Slist_node_base *__head)`
- `template<typename _ToType, typename _FromType >`
`_ToType __static_pointer_cast (_FromType *__arg)`

- `template<typename _ToType, typename _FromType >`
`_ToType __static_pointer_cast (const _FromType &__arg)`
- `size_t __stl_hash_string (const char *__s)`
- `unsigned long __stl_next_prime (unsigned long __n)`
- `template<typename _TRet, typename _Ret = _TRet, typename _CharT, typename... _Base>`
`_Ret __stoa (_TRet(*__convf)(const _CharT *, _CharT **, _Base...), const char *__name, const _CharT *__str,`
`std::size_t *__idx, _Base... __base)`
- `void __throw_concurrency_lock_error ()`
- `void __throw_concurrency_unlock_error ()`
- `void __throw_forced_error ()`
- `template<typename _String, typename _CharT = typename _String::value_type>`
`_String __to_xstring (int(*__convf)(_CharT *, std::size_t, const _CharT *, __builtin_va_list), std::size_t __n,`
`const _CharT *__fmt,...)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter`
`__result)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter`
`__result, std::input_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Size, typename _ForwardIter >`
`std::pair< _RandomAccessIter, _ForwardIter > __uninitialized_copy_n (_RandomAccessIter __first, _Size __count,`
`_ForwardIter __result, std::random_access_iterator_tag)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Allocator >`
`std::pair< _InputIter, _ForwardIter > __uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter`
`__result, _Allocator __alloc)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Tp >`
`std::pair< _InputIter, _ForwardIter > __uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter`
`__result, std::allocator< _Tp >)`
- `void __verbose_terminate_handler ()`
- `std::size_t __Bit_scan_forward (std::size_t __num)`
- `template<typename _ForwardIterator, typename _Allocator >`
`void __Destroy_const (_ForwardIterator __first, _ForwardIterator __last, _Allocator __alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`void __Destroy_const (_ForwardIterator __first, _ForwardIterator __last, std::allocator< _Tp >)`
- `template<class _CharT, class _Traits >`
`void __Rope_fill (std::basic_ostream< _CharT, _Traits > &__o, std::size_t __n)`
- `template<class _CharT >`
`bool __Rope_is_simple (_CharT *)`
- `bool __Rope_is_simple (char *)`
- `bool __Rope_is_simple (wchar_t *)`
- `template<class _Rope_iterator >`
`void __Rope_rotate (_Rope_iterator __first, _Rope_iterator __middle, _Rope_iterator __last)`
- `template<class _CharT >`
`void __S_cond_store_eos (_CharT &)`
- `void __S_cond_store_eos (char &__c)`
- `void __S_cond_store_eos (wchar_t &__c)`
- `template<class _CharT >`
`_CharT __S_eos (_CharT *)`
- `template<class _CharT >`
`bool __S_is_basic_char_type (_CharT *)`
- `bool __S_is_basic_char_type (char *)`
- `bool __S_is_basic_char_type (wchar_t *)`

- `template<class _CharT >`
`bool _S_is_one_byte_char_type (_CharT *)`
- `bool _S_is_one_byte_char_type (char *)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type airy_ai (_Tp __x)`
- `float airy_aif (float __x)`
- `long double airy_ail (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type airy_bi (_Tp __x)`
- `float airy_bif (float __x)`
- `long double airy_bil (long double __x)`
- `template<class _Operation1, class _Operation2 >`
`unary_compose< _Operation1, _Operation2 > compose1 (const _Operation1 &__fn1, const _Operation2 &__fn2)`
- `template<class _Operation1, class _Operation2, class _Operation3 >`
`binary_compose< _Operation1, _Operation2, _Operation3 > compose2 (const _Operation1 &__fn1, const _Operation2 &__fn2, const _Operation3 &__fn3)`
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x)`
- `float conf_hypergf (float __a, float __c, float __x)`
- `long double conf_hypergl (long double __a, long double __c, long double __x)`
- `template<class _Result >`
`constant_void_fun< _Result > constant0 (const _Result &__val)`
- `template<class _Result >`
`constant_unary_fun< _Result, _Result > constant1 (const _Result &__val)`
- `template<class _Result >`
`constant_binary_fun< _Result, _Result, _Result > constant2 (const _Result &__val)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`std::pair< _InputIterator, _OutputIterator > copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`
- `template<typename _InputIterator, typename _Tp, typename _Size >`
`void count (_InputIterator __first, _InputIterator __last, const _Tp &__value, _Size &__n)`
- `template<typename _InputIterator, typename _Predicate, typename _Size >`
`void count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, _Size &__n)`
- `template<typename _InputIterator, typename _Distance >`
`void distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`
- `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)`
- `float hypergf (float __a, float __b, float __c, float __x)`
- `long double hypergl (long double __a, long double __b, long double __c, long double __x)`
- `template<class _Tp >`
`_Tp identity_element (std::multiplies< _Tp >)`
- `template<class _Tp >`
`_Tp identity_element (std::plus< _Tp >)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<class _Ret, class _Tp, class _Arg >`
`std::const_mem_fun1_t< _Ret, _Tp, _Arg > mem_fun1 (_Ret(_Tp::*__f)(_Arg) const)`
- `template<class _Ret, class _Tp, class _Arg >`
`std::mem_fun1_t< _Ret, _Tp, _Arg > mem_fun1 (_Ret(_Tp::*__f)(_Arg))`

- `template<class _Ret, class _Tp, class _Arg >`
`std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun1_ref (_Ret(_Tp::*__f)(_Arg) const)`
- `template<class _Ret, class _Tp, class _Arg >`
`std::mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun1_ref (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator!= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr bool operator!= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _Tp >`
`bool operator!= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool operator!= (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool operator!= (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool operator!= (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator!= (const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator!= (const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >`
`bool operator!= (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`

- `template<class _Tp, class _Alloc >`
`bool operator!= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _Tp >`
`bool operator!= (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Cond >`
`throw_value_base< _Cond > operator* (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (__versa_string< _CharT, _Traits, _Alloc, _Base > &&__lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (__versa_string< _CharT, _Traits, _Alloc, _Base > &&__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (__versa_string< _CharT, _Traits, _Alloc, _Base > &&__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (__versa_string< _CharT, _Traits, _Alloc, _Base > &&__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (_CharT __lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (_CharT __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const _CharT *__lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > operator+ (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator+ (const _Rope_const_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator+ (const _Rope_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > operator+ (const rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > operator+ (const rope< _CharT, _Alloc > &__left, const _CharT *__right)`

- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > operator+ (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__`
`__right)`
- `template<typename _Cond >`
`throw_value_base< _Cond > operator+ (const throw_value_base< _Cond > &__a, const throw_value_base<`
`_Cond > &__b)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator+ (std::ptrdiff_t __n, const _Rope_const_iterator< _CharT,`
`_Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator+ (std::ptrdiff_t __n, const _Rope_iterator< _CharT, _Alloc > &__x)`
- `template<typename _Iterator, typename _Container >`
`constexpr __normal_iterator< _Iterator, _Container > operator+ (typename __normal_iterator< _Iterator, __`
`Container >::difference_type __n, const __normal_iterator< _Iterator, _Container > &__i) noexcept`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & operator+= (rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & operator+= (rope< _CharT, _Alloc > &__left, const _CharT *__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & operator+= (rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<typename _Iterator, typename _Container >`
`constexpr __normal_iterator< _Iterator, _Container >::difference_type operator- (const __normal_iterator< __`
`Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr auto operator- (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator<`
`_IteratorR, _Container > &__rhs) noexcept -> decltype(__lhs.base() - __rhs.base())`
- `template<class _CharT, class _Alloc >`
`std::ptrdiff_t operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< __`
`CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, std::`
`ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`std::ptrdiff_t operator- (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc >`
`&__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator- (const _Rope_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<typename _Cond >`
`throw_value_base< _Cond > operator- (const throw_value_base< _Cond > &__a, const throw_value_base<`
`_Cond > &__b)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator< (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator<`
`_Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator< (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< __`
`IteratorR, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT,`
`_Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const _Pointer_adapter< _Tp1 > & __lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const _Pointer_adapter< _Tp1 > & __lhs, const _Pointer_adapter< _Tp2 > & __rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator< (const _Rope_const_iterator< _CharT, _Alloc > & __x, const _Rope_const_iterator< _CharT, _Alloc > & __y)`
- `template<class _CharT, class _Alloc >`
`bool operator< (const _Rope_iterator< _CharT, _Alloc > & __x, const _Rope_iterator< _CharT, _Alloc > & __y)`
- `template<typename _Value, typename _Int, typename _St >`
`bool operator< (const character< _Value, _Int, _St > & lhs, const character< _Value, _Int, _St > & rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator< (const rope< _CharT, _Alloc > & __left, const rope< _CharT, _Alloc > & __right)`
- `template<class _Tp, class _Alloc >`
`bool operator< (const slist< _Tp, _Alloc > & __SL1, const slist< _Tp, _Alloc > & __SL2)`
- `template<typename _Cond >`
`bool operator< (const throw_value_base< _Cond > & __a, const throw_value_base< _Cond > & __b)`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __o, const rope< _CharT, _Alloc > & __r)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::beta_distribution< _RealType > & __x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::hypergeometric_distribution< _UIntType > & __x)`
- `template<size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > & __x)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::triangular_distribution< _RealType > & __x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::uniform_inside_sphere_distribution< _Dimen, _RealType > & __x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::von_mises_distribution< _RealType > & __x)`
- `template<typename _CharT, typename _Traits, typename _StoreT >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const _Pointer_adapter< _StoreT > & __p)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`arcsine_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`hoyt_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`k_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`logistic_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`nakagami_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`pareto_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`rice_distribution< _RealType > &__x)`
- `std::ostream & operator<< (std::ostream &os, const annotate_base &__b)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator<= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator<`
`_Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator<= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _↵`
`_IteratorR, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _↵`
`_CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _Tp >`
`bool operator<= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT,`
`_Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &↵`
`__y)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _Tp, class _Alloc >`
`bool operator<= (const slist< _Tp, _Alloc > &__SL1, const slist< _Tp, _Alloc > &__SL2)`

- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<size_t _Dimen, typename _RealType >`
`bool operator== (const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d1, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d2)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4>`
`bool operator== (const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__lhs, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__rhs)`
- `template<typename _Tp, typename _Poolp >`
`bool operator== (const __mt_alloc< _Tp, _Poolp > &, const __mt_alloc< _Tp, _Poolp > &)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator== (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr bool operator== (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Tp >`
`bool operator== (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, template< typename, typename, typename > class _Base>`
`__enable_if< std::is_char< _CharT >::value, bool >::type operator== (const __versa_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT >, _Base > &__lhs, const __versa_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT >, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _Tp >`
`bool operator== (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool operator== (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const bitmap_allocator< _Tp1 > &, const bitmap_allocator< _Tp2 > &) throw ()`
- `template<typename _Value, typename _Int, typename _St >`
`bool operator== (const character< _Value, _Int, _St > &__lhs, const character< _Value, _Int, _St > &__rhs)`

- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool operator== (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key,`
`_Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool operator== (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap<`
`_Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator== (const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_multiset<`
`_Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator== (const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_set< _Value,`
`_HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >`
`bool operator== (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key,`
`_HF, _Ex, _Eq, _All > &__ht2)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _Tp, class _Alloc >`
`bool operator== (const slist< _Tp, _Alloc > &__SL1, const slist< _Tp, _Alloc > &__SL2)`
- `template<typename _Tp, typename _Cond >`
`bool operator== (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp, _Cond > &)`
- `template<typename _Cond >`
`bool operator== (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<typename _Tp >`
`bool operator== (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator> (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator<`
`_Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator> (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _↵`
`_IteratorR, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT,`
`_Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _Tp >`
`bool operator> (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT,`
`_Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`

- `template<class _Tp, class _Alloc >`
`bool operator> (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool operator>= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator>= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _Tp >`
`bool operator>= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _Tp, class _Alloc >`
`bool operator>= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::beta_distribution< _RealType > &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::hypergeometric_distribution< _UIntType > &__x)`
- `template<size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__x)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::triangular_distribution< _RealType > &__x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::uniform_inside_sphere_distribution< _Dimen, _RealType > &__x)`

- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::von_mises_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, arcsine_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, hoyt_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, k_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, logistic_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, nakagami_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, pareto_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, rice_distribution< _RealType > & __x)`
- `template<typename _Tp, typename _Integer >`
`_Tp power (_Tp __x, _Integer __n)`
- `template<typename _Tp, typename _Integer, typename _MonoidOperation >`
`_Tp power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`_RandomAccessIterator random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator & __rand)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance >`
`_OutputIterator random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance, typename _RandomNumberGenerator >`
`_OutputIterator random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator & __rand)`
- `void rotate (_Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __first, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __middle, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __last)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`void swap (__versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs)`
- `template<typename _Tp >`
`void swap (_ExtPtr_allocator< _Tp > & __larg, _ExtPtr_allocator< _Tp > & __rarg)`
- `template<class _CharT, class __Alloc >`
`void swap (_Rope_char_ref_proxy< _CharT, __Alloc > __a, _Rope_char_ref_proxy< _CharT, __Alloc > __b)`

- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`void swap (hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`void swap (hash_multimap< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, hash_multimap< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void swap (hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void swap (hash_set< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, hash_set< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _Key, class _HF, class _Extract, class _EqKey, class _All >`
`void swap (hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht1, hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht2)`
- `template<class _CharT, class _Alloc >`
`void swap (rope< _CharT, _Alloc > &__x, rope< _CharT, _Alloc > &__y)`
- `template<class _Tp, class _Alloc >`
`void swap (slist< _Tp, _Alloc > &__x, slist< _Tp, _Alloc > &__y)`
- `template<typename _Cond >`
`void swap (throw_value_base< _Cond > &__a, throw_value_base< _Cond > &__b)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __↵ result)`

Variables

- `const _Lock_policy __default_lock_policy`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > identity_element (_Rope_Concat_fn< _CharT, _Alloc >)`

4.1.1 Detailed Description

GNU extensions for public use.

4.1.2 Typedef Documentation

`__int_traits`

```
template<typename _Tp >
using __gnu_cxx::__int_traits = typedef __numeric_traits_integer<_Tp>
Convenience alias for __numeric_traits<integer-type>.
```

4.1.3 Function Documentation

`__static_pointer_cast()` [1/2]

```
template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__static_pointer_cast (
    _FromType * __arg ) [inline]
```

Casting operations for cases where `_FromType` is a standard pointer. `_ToType` can be a standard or non-standard pointer.

__static_pointer_cast() [2/2]

```
template<typename _ToType , typename _FromType >
_ToType __gnu_cxx::__static_pointer_cast (
    const _FromType & __arg ) [inline]
```

Casting operations for cases where `_FromType` is not a standard pointer. `_ToType` can be a standard or non-standard pointer. Given that `_FromType` is not a pointer, it must have a `get()` method that returns the standard pointer equivalent of the address it points to, and must have an `element_type` typedef which names the type it points to.

_Bit_scan_forward()

```
std::size_t __gnu_cxx::_Bit_scan_forward (
    std::size_t __num ) [inline]
```

Generic Version of the `bsf` instruction.

Referenced by [__gnu_cxx::bitmap_allocator<_Tp>::__M_allocate_single_object\(\)](#).

operator"!=() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator!= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test difference of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

operator"!=() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator!= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test difference of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

operator"!="() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator!= (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test difference of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) != 0`. False otherwise.

operator+() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (
    _CharT __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs )
```

Concatenate character and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::reserve\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator+\(\)](#).

operator+() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    _CharT __rhs )
```

Concatenate string and character.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::reserve\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size\(\)](#).

operator+() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs )
```

Concatenate two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::reserve\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size\(\)](#).

operator+() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs )
```

Concatenate string and C string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

operator+() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs )
```

Concatenate C string and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

operator<() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator< (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if string precedes string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#).

operator<() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator< (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string precedes C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator<() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator< (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if C string precedes string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator<=() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator<= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if string doesn't follow string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator<=() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
bool __gnu_cxx::operator<= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string doesn't follow C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator<=() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator<= (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if C string doesn't follow string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator==() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator== (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test equivalence of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator==() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator== (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test equivalence of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator==() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator== (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test equivalence of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) == 0`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator==() [4/4]

```
template<typename _Tp >
bool __gnu_cxx::operator== (
    const _Pointer_adapter< _Tp > & __lhs,
    const _Pointer_adapter< _Tp > & __rhs ) [inline]
```

Comparison operators for `_Pointer_adapter` defer to the base class' comparison operators, when possible.

operator>() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator> (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if string follows string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator>() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator> (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string follows C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#).

operator>() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator> (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if C string follows string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#).

operator>=() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator>= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if string doesn't precede string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#).

operator>=() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator>= (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string doesn't precede C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#).

operator>=() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::operator>= (
    const _CharT * __lhs,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Test if C string doesn't precede string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#).

swap()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::swap (
    __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs,
    __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs ) [inline]
```

Swap contents of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Exchanges the contents of `__lhs` and `__rhs` in constant time.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::swap\(\)](#).

4.2 `__gnu_cxx::__detail` Namespace Reference

Classes

- class [__mini_vector](#)
- class [_Bitmap_counter](#)
- class [_Ffit_finder](#)

Enumerations

- enum { `_S_max_rope_depth` }
- enum { `bits_per_byte` , `bits_per_block` }
- enum `_Tag` { `_S_leaf` , `_S_concat` , `_S_substringfn` , `_S_function` }

Functions

- void [__bit_allocate](#) (std::size_t *__pmap, std::size_t __pos) throw ()
- void [__bit_free](#) (std::size_t *__pmap, std::size_t __pos) throw ()
- template<typename _ForwardIterator , typename _Tp , typename _Compare >
_ForwardIterator **__lower_bound** (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)
- template<typename _AddrPair >
std::size_t [__num_bitmaps](#) (_AddrPair __ap)
- template<typename _AddrPair >
std::size_t [__num_blocks](#) (_AddrPair __ap)

4.2.1 Detailed Description

Implementation details not part of the namespace `__gnu_cxx` interface.

4.2.2 Function Documentation

`__bit_allocate()`

```
void __gnu_cxx::__detail::__bit_allocate (
    std::size_t * __pmap,
    std::size_t __pos ) throw ( )    [inline]
```

Mark a memory address as allocated by re-setting the corresponding bit in the bit-map.

Referenced by [__gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object\(\)](#).

`__bit_free()`

```
void __gnu_cxx::__detail::__bit_free (
    std::size_t * __pmap,
    std::size_t __pos ) throw ( )    [inline]
```

Mark a memory address as free by setting the corresponding bit in the bit-map.

Referenced by [__gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object\(\)](#).

`__num_bitmaps()`

```
template<typename _AddrPair >
std::size_t __gnu_cxx::__detail::__num_bitmaps (
    _AddrPair __ap ) [inline]
```

The number of Bit-maps pointed to by the address pair passed to the function.

References [__num_blocks\(\)](#).

Referenced by [__gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object\(\)](#), and [__gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object\(\)](#).

`__num_blocks()`

```
template<typename _AddrPair >
std::size_t __gnu_cxx::__detail::__num_blocks (
    _AddrPair __ap ) [inline]
```

The number of Blocks pointed to by the address pair passed to the function.

Referenced by [__num_bitmaps\(\)](#).

4.3 `__gnu_cxx::typelist` Namespace Reference

Functions

- `template<typename Fn , typename Typelist >`
`void apply (Fn &, Typelist)`
- `template<typename Fn , typename Typelist >`
`void apply_generator (Fn &fn, Typelist)`
- `template<typename Fn , typename TypelistT , typename TypelistV >`
`void apply_generator (Fn &fn, TypelistT, TypelistV)`
- `template<typename Gn , typename Typelist >`
`void apply_generator (Gn &, Typelist)`
- `template<typename Gn , typename TypelistT , typename TypelistV >`
`void apply_generator (Gn &, TypelistT, TypelistV)`

4.3.1 Detailed Description

GNU typelist extensions for public compile-time use.

4.3.2 Function Documentation

apply_generator()

```
template<typename Gn , typename Typelist >
void __gnu_cxx::typelist::apply_generator (
    Gn & ,
    Typelist )
```

Apply all typelist types to generator functor.

4.4 __gnu_debug Namespace Reference

Classes

- class [_After_nth_from](#)
- struct [_BeforeBeginHelper](#)
- class [_Equal_to](#)
- class [_Not_equal_to](#)
- class [_Safe_container](#)
- class [_Safe_forward_list](#)
- class [_Safe_iterator](#)
- class [_Safe_iterator_base](#)
- class [_Safe_local_iterator](#)
- class [_Safe_local_iterator_base](#)
- class [_Safe_node_sequence](#)
- class [_Safe_sequence](#)
- class [_Safe_sequence_base](#)
- class [_Safe_unordered_container](#)
- class [_Safe_unordered_container_base](#)
- class [_Safe_vector](#)
- struct [_Sequence_traits](#)
- class [basic_string](#)

Typedefs

- typedef [basic_string](#)< char > **string**
- typedef [basic_string](#)< char16_t > **u16string**
- typedef [basic_string](#)< char32_t > **u32string**
- typedef [basic_string](#)< wchar_t > **wstring**

Enumerations

- enum **_Debug_msg_id** {
[__msg_valid_range](#) , [__msg_insert_singular](#) , [__msg_insert_different](#) , [__msg_erase_bad](#) ,
[__msg_erase_different](#) , [__msg_subscript_oob](#) , [__msg_empty](#) , [__msg_unpartitioned](#) ,
[__msg_unpartitioned_pred](#) , [__msg_unsorted](#) , [__msg_unsorted_pred](#) , [__msg_not_heap](#) ,
[__msg_not_heap_pred](#) , [__msg_bad_bitset_write](#) , [__msg_bad_bitset_read](#) , [__msg_bad_bitset_flip](#) ,
[__msg_self_splice](#) , [__msg_splice_alloc](#) , [__msg_splice_bad](#) , [__msg_splice_other](#) ,
[__msg_splice_overlap](#) , [__msg_init_singular](#) , [__msg_init_copy_singular](#) , [__msg_init_const_singular](#) ,
[__msg_copy_singular](#) , [__msg_bad_deref](#) , [__msg_bad_inc](#) , [__msg_bad_dec](#) ,


```

__msg_iter_subscript_oob , __msg_advance_oob , __msg_retreat_oob , __msg_iter_compare_bad ,
__msg_compare_different , __msg_iter_order_bad , __msg_order_different , __msg_distance_bad ,
__msg_distance_different , __msg_deref_istream , __msg_inc_istream , __msg_output_ostream ,
__msg_deref_istreambuf , __msg_inc_istreambuf , __msg_insert_after_end , __msg_erase_after_bad ,
__msg_valid_range2 , __msg_local_iter_compare_bad , __msg_non_empty_range , __msg_self_move↵
assign ,
__msg_bucket_index_oob , __msg_valid_load_factor , __msg_equal_allocs , __msg_insert_range↵
from_self ,
__msg_irreflexive_ordering }
• enum \_Distance\_precision {
    __dp_none , __dp_equality , __dp_sign , __dp_sign_max_size ,
    __dp_exact }

```

Functions

```

• template<typename _Iterator >
  constexpr _Iterator \_\_base (_Iterator __it)
• template<typename _Iterator , typename _Sequence >
  _Iterator \_\_base (const \_Safe\_iterator< _Iterator, _Sequence, std::random\_access\_iterator\_tag > &__it)
• template<typename _Iterator >
  constexpr auto \_\_base (const std::move\_iterator< _Iterator > &__it) -> decltype(std::make_move_iterator(__↵
base(__it.base())))
• template<typename _Iterator , typename _Sequence >
  constexpr std::reverse\_iterator< _Iterator > \_\_base (const std::reverse\_iterator< \_Safe\_iterator< _Iterator, _↵
Sequence, std::random\_access\_iterator\_tag > > &__it)
• template<typename _InputIterator , typename _Size >
  constexpr bool \_\_can\_advance (_InputIterator, _Size)
• template<typename _InputIterator , typename _Diff >
  constexpr bool \_\_can\_advance (_InputIterator, const std::pair< _Diff, \_Distance\_precision > &, int)
• template<typename _Iterator , typename _Sequence , typename _Category , typename _Size >
  bool \_\_can\_advance (const \_Safe\_iterator< _Iterator, _Sequence, _Category > &, _Size)
• template<typename _Iterator , typename _Sequence , typename _Category , typename _Diff >
  bool \_\_can\_advance (const \_Safe\_iterator< _Iterator, _Sequence, _Category > &, const std::pair< _Diff,
\_Distance\_precision > &, int)
• template<typename _Iterator , typename _Size >
  constexpr bool \_\_can\_advance (const std::move\_iterator< _Iterator > &__it, _Size __n)
• template<typename _Iterator , typename _Diff >
  constexpr bool \_\_can\_advance (const std::move\_iterator< _Iterator > &__it, const std::pair< _Diff,
\_Distance\_precision > &__dist, int __way)
• template<typename _Iterator , typename _Size >
  constexpr bool \_\_can\_advance (const std::reverse\_iterator< _Iterator > &__it, _Size __n)
• template<typename _Iterator , typename _Diff >
  constexpr bool \_\_can\_advance (const std::reverse\_iterator< _Iterator > &__it, const std::pair< _Diff,
\_Distance\_precision > &__dist, int __way)
• template<typename _ForwardIterator , typename _Tp >
  constexpr bool \_\_check\_partitioned\_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵
value)
• template<typename _ForwardIterator , typename _Tp , typename _Pred >
  constexpr bool \_\_check\_partitioned\_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵
value, _Pred __pred)
• template<typename _ForwardIterator , typename _Tp >
  constexpr bool \_\_check\_partitioned\_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵
value)

```

- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`constexpr bool __check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value, _Pred __pred)`
- `template<typename _Iterator >`
`constexpr bool __check_singular (_Iterator const &)`
- `template<typename _Tp >`
`constexpr bool __check_singular (_Tp *const &__ptr)`
- `bool __check_singular_aux (const _Safe_iterator_base * __x)`
- `bool __check_singular_aux (const void *)`
- `template<typename _InputIterator >`
`constexpr bool __check_sorted (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __check_sorted (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr bool __check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, std::forward_iterator_tag)`
- `template<typename _ForwardIterator >`
`constexpr bool __check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __check_sorted_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::input_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __check_sorted_aux (const _InputIterator &, const _InputIterator &, std::input_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr bool __check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Predicate >`
`constexpr bool __check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __check_sorted_set_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::__false_type)`
- `template<typename _InputIterator >`
`constexpr bool __check_sorted_set_aux (const _InputIterator &, const _InputIterator &, std::__false_type)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred, std::__true_type)`
- `template<typename _InputIterator >`
`constexpr bool __check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, std::__true_type)`
- `template<typename _CharT, typename _Integer >`
`const _CharT * __check_string (const _CharT * __s, _Integer __n, const char * __file, unsigned int __line, const char * __function)`
- `template<typename _CharT >`
`const _CharT * __check_string (const _CharT * __s, const char * __file, unsigned int __line, const char * __function)`
- `template<typename _InputIterator >`
`_InputIterator __check_valid_range (const _InputIterator &__first, const _InputIterator &__last, const char * __file, unsigned int __line, const char * __function)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __foreign_iterator (const _Safe_iterator< _Iterator, _Sequence, _Category > &__it, _InputIterator __other, _InputIterator __other_end)`

- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Integral >`
`bool __foreign_iterator_aux (const __Safe_iterator< _Iterator, _Sequence, _Category > &, _Integral, _Integral,`
`std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __foreign_iterator_aux (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, _InputIterator`
`__other, _InputIterator __other_end, std::__false_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _OtherIterator, typename _OtherSequence, type-`
`name _OtherCategory >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &, const __Safe_iterator<`
`_OtherIterator, _OtherSequence, _OtherCategory > &, const __Safe_iterator< _OtherIterator, _OtherSequence,`
`_OtherCategory > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const _Input`
`Iterator &__other, const _InputIterator &__other_end)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _OtherIterator >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const`
`__Safe_iterator< _OtherIterator, _Sequence, _Category > &__other, const __Safe_iterator< _OtherIterator,`
`_Sequence, _Category > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence, _Category > &, const _InputIterator`
`&, const _InputIterator &, std::__false_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const _Input`
`Iterator &__other, const _InputIterator &__other_end, std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence, _Category > &,...)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const typename`
`_Sequence::value_type *__other)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __get_distance (_Iterator __lhs, _Iterator __rhs)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __get_distance (_Iterator __lhs, _Iterator __rhs, std::input_iterator_tag)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __get_distance (_Iterator __lhs, _Iterator __rhs, std::random_access_iterator_tag)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __get_distance (const std::move_iterator< _Iterator > &__first,`
`const std::move_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __get_distance (const std::reverse_iterator< _Iterator > &__`
`__first, const std::reverse_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`constexpr bool __is_irreflexive (_Iterator __it)`
- `template<typename _Iterator, typename _Pred >`
`constexpr bool __is_irreflexive_pred (_Iterator __it, _Pred __pred)`
- `template<typename _Iterator >`
`_Iterator __unsafe (_Iterator __it)`
- `template<typename _Iterator, typename _Sequence >`
`_Iterator __unsafe (const __Safe_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator, typename _Sequence >`
`_Iterator __unsafe (const __Safe_local_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator >`
`constexpr auto __unsafe (const std::move_iterator< _Iterator > &__it) -> decltype(std::make_move_iterator(`
`__unsafe(__it.base())))`

- `template<typename _Iterator >`
`constexpr auto __unsafe (const std::reverse_iterator< _Iterator > &__it) -> decltype(std::__make_reverse_iterator(__unsafe(__it.base())))`
- `template<typename _InputIterator >`
`constexpr bool __valid_range (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator >`
`constexpr bool __valid_range (_InputIterator __first, _InputIterator __last, typename _Distance_traits< _InputIterator >::__type &__dist)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __valid_range (const _Safe_iterator< _Iterator, _Sequence, _Category > &, const _Safe_iterator< _Iterator, _Sequence, _Category > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __valid_range (const _Safe_iterator< _Iterator, _Sequence, _Category > &, const _Safe_iterator< _Iterator, _Sequence, _Category > &, typename _Distance_traits< _Iterator >::__type &)`
- `template<typename _Iterator, typename _Sequence >`
`bool __valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator< _Iterator, _Sequence > &)`
- `template<typename _Iterator, typename _Sequence >`
`bool __valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator< _Iterator, _Sequence > &, typename _Distance_traits< _Iterator >::__type &)`
- `template<typename _Iterator >`
`constexpr bool __valid_range (const std::move_iterator< _Iterator > &__first, const std::move_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _Iterator >`
`constexpr bool __valid_range (const std::reverse_iterator< _Iterator > &__first, const std::reverse_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _InputIterator >`
`constexpr bool __valid_range_aux (_InputIterator __first, _InputIterator __last, std::__false_type)`
- `template<typename _InputIterator >`
`constexpr bool __valid_range_aux (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __valid_range_aux (_InputIterator __first, _InputIterator __last, std::random_access_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __valid_range_aux (_InputIterator __first, _InputIterator __last, typename _Distance_traits< _InputIterator >::__type &__dist, std::__false_type)`
- `template<typename _Integral >`
`constexpr bool __valid_range_aux (_Integral, _Integral, std::__true_type)`
- `template<typename _Integral >`
`constexpr bool __valid_range_aux (_Integral, _Integral, typename _Distance_traits< _Integral >::__type &__dist, std::__true_type)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > & getline (std::basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > & getline (std::basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Allocator > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`

- Generated by Doxygen

- `template<typename _CharT, typename _Traits, typename _Allocator >
std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,
basic_string< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _CharT, typename _Traits, typename _Allocator >
void swap (basic_string< _CharT, _Traits, _Allocator > &__lhs, basic_string< _CharT, _Traits, _Allocator > &__rhs)`

4.4.1 Detailed Description

GNU debug classes for public use.

4.4.2 Typedef Documentation

`u16string`

typedef `basic_string<char16_t> __gnu_debug::u16string`
A string of `char16_t`.

`u32string`

typedef `basic_string<char32_t> __gnu_debug::u32string`
A string of `char32_t`.

4.4.3 Enumeration Type Documentation

`__Distance_precision`

enum `__gnu_debug::__Distance_precision`

The precision to which we can calculate the distance between two iterators.

4.4.4 Function Documentation

`__base()`

```
template<typename _Iterator >
constexpr _Iterator __gnu_debug::__base (
    _Iterator __it ) [inline], [constexpr]
```

Helper function to extract base iterator of random access safe iterator in order to reduce performance impact of debug mode. Limited to random access iterator because it is the only category for which it is possible to check for correct iterators order in the `__valid_range` function thanks to the `<` operator.

Referenced by `__gnu_debug::__Safe_iterator< _Iterator, _Sequence, _Category >::__M_before_dereferenceable()`.

`__check_singular()`

```
template<typename _Tp >
constexpr bool __gnu_debug::__check_singular (
    _Tp *const & __ptr ) [inline], [constexpr]
```

Non-NULL pointers are nonsingular.

`__check_singular_aux()`

```
bool __gnu_debug::__check_singular_aux (
    const __Safe_iterator_base * __x ) [inline]
```

Iterators that derive from `__Safe_iterator_base` can be determined singular or non-singular.

References `__gnu_debug::__Safe_iterator_base::__M_singular()`.

__check_string() [1/2]

```
template<typename _CharT , typename _Integer >
const _CharT * __gnu_debug::__check_string (
    const _CharT * __s,
    _Integer __n,
    const char * __file,
    unsigned int __line,
    const char * __function ) [inline]
```

Checks that `__s` is non-NULL or `__n == 0`, and then returns `__s`.

__check_string() [2/2]

```
template<typename _CharT >
const _CharT * __gnu_debug::__check_string (
    const _CharT * __s,
    const char * __file,
    unsigned int __line,
    const char * __function ) [inline]
```

Checks that `__s` is non-NULL and then returns `__s`.

__foreign_iterator_aux2() [1/2]

```
template<typename _Iterator , typename _Sequence , typename _Category , typename _OtherIterator ,
typename _OtherSequence , typename _OtherCategory >
bool __gnu_debug::__foreign_iterator_aux2 (
    const _Safe_iterator< _Iterator, _Sequence, _Category > & ,
    const _Safe_iterator< _OtherIterator, _OtherSequence, _OtherCategory > & ,
    const _Safe_iterator< _OtherIterator, _OtherSequence, _OtherCategory > & ) [inline]
```

Handle debug iterators from different types of container.

__foreign_iterator_aux2() [2/2]

```
template<typename _Iterator , typename _Sequence , typename _Category , typename _OtherIterator >
bool __gnu_debug::__foreign_iterator_aux2 (
    const _Safe_iterator< _Iterator, _Sequence, _Category > & __it,
    const _Safe_iterator< _OtherIterator, _Sequence, _Category > & __other,
    const _Safe_iterator< _OtherIterator, _Sequence, _Category > & ) [inline]
```

Handle debug iterators from the same type of container.

__get_distance()

```
template<typename _Iterator >
constexpr _Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (
    _Iterator __lhs,
    _Iterator __rhs,
    std::random_access_iterator_tag ) [inline], [constexpr]
```

Determine the distance between two iterators with some known precision.

__valid_range() [1/3]

```
template<typename _InputIterator >
constexpr bool __gnu_debug::__valid_range (
    _InputIterator __first,
```

```

    _InputIterator __last,
    typename _Distance_traits< _InputIterator >::__type & __dist ) [inline], [constexpr]

```

Don't know what these iterators are, or if they are even iterators (we may get an integral type for `InputIterator`), so see if they are integral and pass them on to the next phase otherwise.

References [__valid_range_aux\(\)](#).

`__valid_range()` [2/3]

```

template<typename _Iterator , typename _Sequence , typename _Category >
bool __gnu_debug::__valid_range (
    const _Safe_iterator< _Iterator, _Sequence, _Category > & __first,
    const _Safe_iterator< _Iterator, _Sequence, _Category > & __last,
    typename _Distance_traits< _Iterator >::__type & __dist ) [inline]

```

Safe iterators know how to check if they form a valid range.

`__valid_range()` [3/3]

```

template<typename _Iterator , typename _Sequence >
bool __gnu_debug::__valid_range (
    const _Safe_local_iterator< _Iterator, _Sequence > & __first,
    const _Safe_local_iterator< _Iterator, _Sequence > & __last,
    typename _Distance_traits< _Iterator >::__type & __dist_info ) [inline]

```

Safe local iterators know how to check if they form a valid range.

`__valid_range_aux()` [1/2]

```

template<typename _InputIterator >
constexpr bool __gnu_debug::__valid_range_aux (
    _InputIterator __first,
    _InputIterator __last,
    std::__false_type ) [inline], [constexpr]

```

We have iterators, so figure out what kind of iterators they are to see if we can check the range ahead of time.

References [std::__iterator_category\(\)](#), and [__valid_range_aux\(\)](#).

`__valid_range_aux()` [2/2]

```

template<typename _Integral >
constexpr bool __gnu_debug::__valid_range_aux (
    _Integral ,
    _Integral ,
    std::__true_type ) [inline], [constexpr]

```

We say that integral types for a valid range, and defer to other routines to realize what to do with integral types instead of iterators.

Referenced by [__valid_range\(\)](#), and [__valid_range_aux\(\)](#).

4.5 `__gnu_internal` Namespace Reference

4.5.1 Detailed Description

GNU implementation details, not for public use or export. Used only when anonymous namespaces cannot be substituted.

4.6 `__gnu_parallel` Namespace Reference

Classes

- struct [__accumulate_binop_reduct](#)
- struct [__accumulate_selector](#)
- struct [__adjacent_difference_selector](#)
- struct [__adjacent_find_selector](#)
- class [__binder1st](#)
- class [__binder2nd](#)
- struct [__count_if_selector](#)
- struct [__count_selector](#)
- struct [__fill_selector](#)
- struct [__find_first_of_selector](#)
- struct [__find_if_selector](#)
- struct [__for_each_selector](#)
- struct [__generate_selector](#)
- struct [__generic_find_selector](#)
- struct [__generic_for_each_selector](#)
- struct [__identity_selector](#)
- struct [__inner_product_selector](#)
- struct [__max_element_reduct](#)
- struct [__min_element_reduct](#)
- struct [__mismatch_selector](#)
- struct [__multiway_merge_3_variant_sentinel_switch](#)
- struct [__multiway_merge_3_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__multiway_merge_4_variant_sentinel_switch](#)
- struct [__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__multiway_merge_k_variant_sentinel_switch](#)
- struct [__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__replace_if_selector](#)
- struct [__replace_selector](#)
- struct [__transform1_selector](#)
- struct [__transform2_selector](#)
- class [__unary_negate](#)
- struct [_DRandomShufflingGlobalData](#)
- struct [_DRSSorterPU](#)
- struct [_DummyReduct](#)
- class [_EqualFromLess](#)
- struct [_EqualTo](#)
- class [_GuardedIterator](#)
- class [_IteratorPair](#)
- class [_IteratorTriple](#)
- struct [_Job](#)
- struct [_Less](#)
- class [_Lexicographic](#)
- class [_LexicographicReverse](#)
- class [_LoserTree](#)
- class [_LoserTree< false, _Tp, _Compare >](#)
- class [_LoserTreeBase](#)
- class [_LoserTreePointer](#)
- class [_LoserTreePointer< false, _Tp, _Compare >](#)

- class [_LoserTreePointerBase](#)
- class [_LoserTreePointerUnguarded](#)
- class [_LoserTreePointerUnguarded< false, _Tp, _Compare >](#)
- class [_LoserTreePointerUnguardedBase](#)
- struct [_LoserTreeTraits](#)
- class [_LoserTreeUnguarded](#)
- class [_LoserTreeUnguarded< false, _Tp, _Compare >](#)
- class [_LoserTreeUnguardedBase](#)
- struct [_Multiplies](#)
- struct [_Nothing](#)
- struct [_Piece](#)
- struct [_Plus](#)
- struct [_PMWMSSortingData](#)
- class [_PseudoSequence](#)
- class [_PseudoSequenceIterator](#)
- struct [_QSBThreadLocal](#)
- class [_RandomNumber](#)
- class [_RestrictedBoundedConcurrentQueue](#)
- struct [_SamplingSorter](#)
- struct [_SamplingSorter< false, _RAIter, _StrictWeakOrdering >](#)
- struct [_Settings](#)
- struct [_SplitConsistently](#)
- struct [_SplitConsistently< false, _RAIter, _Compare, _SortingPlacesIterator >](#)
- struct [_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIterator >](#)
- struct [balanced_quicksort_tag](#)
- struct [balanced_tag](#)
- struct [constant_size_blocks_tag](#)
- struct [default_parallel_tag](#)
- struct [equal_split_tag](#)
- struct [exact_tag](#)
- struct [find_tag](#)
- struct [growing_blocks_tag](#)
- struct [multiway_mergesort_exact_tag](#)
- struct [multiway_mergesort_sampling_tag](#)
- struct [multiway_mergesort_tag](#)
- struct [omp_loop_static_tag](#)
- struct [omp_loop_tag](#)
- struct [parallel_tag](#)
- struct [quicksort_tag](#)
- struct [sampling_tag](#)
- struct [sequential_tag](#)
- struct [unbalanced_tag](#)

Typedefs

- typedef unsigned short [_BinIndex](#)
- typedef int64_t [_CASable](#)
- typedef uint64_t [_SequenceIndex](#)
- typedef uint16_t [_ThreadIndex](#)

Enumerations

- enum `_AlgorithmStrategy` { `heuristic` , `force_sequential` , `force_parallel` }
- enum `_FindAlgorithm` { `GROWING_BLOCKS` , `CONSTANT_SIZE_BLOCKS` , `EQUAL_SPLIT` }
- enum `_MultiwayMergeAlgorithm` { `LOSER_TREE` }
- enum `_Parallelism` {
 `sequential` , `parallel_unbalanced` , `parallel_balanced` , `parallel_omp_loop` ,
 `parallel_omp_loop_static` , `parallel_taskqueue` }
- enum `_PartialSumAlgorithm` { `RECURSIVE` , `LINEAR` }
- enum `_SortAlgorithm` { `MWMS` , `QS` , `QS_BALANCED` }
- enum `_SplittingAlgorithm` { `SAMPLING` , `EXACT` }

Functions

- template<typename `_Tp` >
 `_Tp __add_omp` (volatile `_Tp *``__ptr`, `_Tp` `__addend`)
- template<typename `_RAlter` , typename `_DifferenceTp` >
 void `__calc_borders` (`_RAlter` `__elements`, `_DifferenceTp` `__length`, `_DifferenceTp *``__off`)
- template<typename `_Tp` >
 bool `__cas_omp` (volatile `_Tp *``__ptr`, `_Tp` `__comparand`, `_Tp` `__replacement`)
- template<typename `_Tp` >
 bool `__compare_and_swap` (volatile `_Tp *``__ptr`, `_Tp` `__comparand`, `_Tp` `__replacement`)
- template<typename `_Iter` , typename `_OutputIterator` >
 `_OutputIterator` `__copy_tail` (`std::pair`< `_Iter`, `_Iter` > `__b`, `std::pair`< `_Iter`, `_Iter` > `__e`, `_OutputIterator` `__r`)
- void `__decode2` (`_CASable` `__x`, int &`__a`, int &`__b`)
- template<typename `_RAlter` , typename `_DifferenceTp` >
 void `__determine_samples` (`_PMWMSortingData`< `_RAlter` > `*__sd`, `_DifferenceTp` `__num_samples`)
- `_CASable` `__encode2` (int `__a`, int `__b`)
- template<typename `_DifferenceType` , typename `_OutputIterator` >
 `_OutputIterator` `__equally_split` (`_DifferenceType` `__n`, `_ThreadIndex` `__num_threads`, `_OutputIterator` `__s`)
- template<typename `_DifferenceType` >
 `_DifferenceType` `__equally_split_point` (`_DifferenceType` `__n`, `_ThreadIndex` `__num_threads`, `_ThreadIndex` `__thread_no`)
- template<typename `_Tp` >
 `_Tp` `__fetch_and_add` (volatile `_Tp *``__ptr`, `_Tp` `__addend`)
- template<typename `_RAlter1` , typename `_RAlter2` , typename `_Pred` , typename `_Selector` >
 `std::pair`< `_RAlter1`, `_RAlter2` > `__find_template` (`_RAlter1` `__begin1`, `_RAlter1` `__end1`, `_RAlter2` `__begin2`, `_Pred` `__pred`, `_Selector` `__selector`)
- template<typename `_RAlter1` , typename `_RAlter2` , typename `_Pred` , typename `_Selector` >
 `std::pair`< `_RAlter1`, `_RAlter2` > `__find_template` (`_RAlter1` `__begin1`, `_RAlter1` `__end1`, `_RAlter2` `__begin2`, `_Pred` `__pred`, `_Selector` `__selector`, `constant_size_blocks_tag`)
- template<typename `_RAlter1` , typename `_RAlter2` , typename `_Pred` , typename `_Selector` >
 `std::pair`< `_RAlter1`, `_RAlter2` > `__find_template` (`_RAlter1` `__begin1`, `_RAlter1` `__end1`, `_RAlter2` `__begin2`, `_Pred` `__pred`, `_Selector` `__selector`, `equal_split_tag`)
- template<typename `_RAlter1` , typename `_RAlter2` , typename `_Pred` , typename `_Selector` >
 `std::pair`< `_RAlter1`, `_RAlter2` > `__find_template` (`_RAlter1` `__begin1`, `_RAlter1` `__end1`, `_RAlter2` `__begin2`, `_Pred` `__pred`, `_Selector` `__selector`, `growing_blocks_tag`)
- template<typename `_Iter` , typename `_UserOp` , typename `_Functionality` , typename `_Red` , typename `_Result` >
 `_UserOp` `__for_each_template_random_access` (`_Iter` `__begin`, `_Iter` `__end`, `_UserOp` `__user_op`, `_Functionality` & `__functionality`, `_Red` `__reduction`, `_Result` `__reduction_start`, `_Result` & `__output`, typename `std::iterator_traits`< `_Iter` >::`difference_type` `__bound`, `_Parallelism` `__parallelism_tag`)
- template<typename `_RAlter` , typename `_Op` , typename `_Fu` , typename `_Red` , typename `_Result` >
 `_Op` `__for_each_template_random_access_ed` (`_RAlter` `__begin`, `_RAlter` `__end`, `_Op` `__o`, `_Fu` & `__f`, `_Red` `__r`, `_Result` `__base`, `_Result` & `__output`, typename `std::iterator_traits`< `_RAlter` >::`difference_type` `__bound`)

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_omp_loop (_RAIter __begin, _RAIter __end, _Op __o, _Fu &__f, ↵`
`_Red __r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >::difference_type __bound)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_omp_loop_static (_RAIter __begin, _RAIter __end, _Op __o, _Fu`
`&__f, _Red __r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >::difference_type`
`__bound)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_workstealing (_RAIter __begin, _RAIter __end, _Op __op, _Fu &↵`
`__f, _Red __r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >::difference_type`
`__bound)`
- `__ThreadIndex __get_max_threads ()`
- `bool __is_parallel (const __Parallelism __p)`
- `template<typename _Iter, typename _Compare >`
`bool __is_sorted (_Iter __begin, _Iter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter __median_of_three_iterators (_RAIter __a, _RAIter __b, _RAIter __c, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 ↵`
`__end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance_movc (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2`
`__end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance_usual (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2`
`__end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter3, typename _Compare >`
`_RAIter3 __parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter1 &__begin2, _RAIter1 ↵`
`__end2, _RAIter3 __target, typename std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare`
`__comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _Compare >`
`_RAIter3 __parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 ↵`
`__end2, _RAIter3 __target, typename std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare`
`__comp)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation`
`__bin_op)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum_basecase (_Iter __begin, _Iter __end, _OutputIterator __result, ↵`
`_BinaryOperation __bin_op, typename std::iterator_traits< _Iter >::value_type __value)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum_linear (_Iter __begin, _Iter __end, _OutputIterator __result, _Binary↵`
`Operation __bin_op, typename std::iterator_traits< _Iter >::difference_type __n)`
- `template<typename _RAIter, typename _Predicate >`
`std::iterator_traits< _RAIter >::difference_type __parallel_partition (_RAIter __begin, _RAIter __end, _Predicate`
`__pred, __ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator __↵`
`rng=__RandomNumber())`

- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle_drs (_RAIter __begin, _RAIter __end, typename std::iterator_traits<_RAIter>::difference_type __n, _ThreadIndex __num_threads, _RandomNumberGenerator &__rng)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle_drs_pu (_DRSSorterPU<_RAIter, _RandomNumberGenerator> *__pus)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_intersection (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Operation >`
`_OutputIterator __parallel_set_operation (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Operation __op)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_symmetric_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_union (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<bool __stable, typename _RAIter, typename _Compare, typename _Parallelism >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, balanced_quicksort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, default_parallel_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_exact_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_sampling_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, parallel_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, quicksort_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qs (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qs_conquer (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits<_RAIter>::difference_type __parallel_sort_qs_divide (_RAIter __begin, _RAIter __end, _Compare __comp, typename std::iterator_traits<_RAIter>::difference_type __pivot_rank, typename std::iterator_traits<_RAIter>::difference_type __num_samples, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qsb (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _Iter, class _OutputIterator >`
`_OutputIterator __parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result)`

- `template<typename _Iter, class _OutputIterator, class _BinaryPredicate >`
`_OutputIterator __parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result, _BinaryPredicate`
`__binary_pred)`
- `template<typename _RAlter, typename _Compare >`
`void __qsb_conquer (_QSBThreadLocal< _RAlter > **__tls, _RAlter __begin, _RAlter __end, _Compare __`
`comp, _ThreadIndex __iam, _ThreadIndex __num_threads, bool __parent_wait)`
- `template<typename _RAlter, typename _Compare >`
`std::iterator_traits< _RAlter >::difference_type __qsb_divide (_RAlter __begin, _RAlter __end, _Compare __`
`comp, _ThreadIndex __num_threads)`
- `template<typename _RAlter, typename _Compare >`
`void __qsb_local_sort_with_helping (_QSBThreadLocal< _RAlter > **__tls, _Compare &__comp, _ThreadIndex`
`__iam, bool __wait)`
- `template<typename _RandomNumberGenerator >`
`int __random_number_pow2 (int __logp, _RandomNumberGenerator &__rng)`
- `template<typename _Size >`
`_Size __rd_log2 (_Size __n)`
- `template<typename _Tp >`
`_Tp __round_up_to_pow2 (_Tp __x)`
- `template<typename __RAlter1, typename __RAlter2, typename _Pred >`
`__RAlter1 __search_template (__RAlter1 __begin1, __RAlter1 __end1, __RAlter2 __begin2, __RAlter2 __end2,`
`_Pred __pred)`
- `template<bool __stable, bool __sentinels, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp, typename _Compare`
`>`
`_RAlter3 __sequential_multiway_merge (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _RAlter3`
`__target, const typename std::iterator_traits< typename std::iterator_traits< _RAlterIterator >::value_type`
`<::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void __sequential_random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &__rng)`
- `template<typename _Iter >`
`void __shrink (std::vector< _Iter > &__os_starts, size_t &__count_to_two, size_t &__range_length)`
- `template<typename _Iter >`
`void __shrink_and_double (std::vector< _Iter > &__os_starts, size_t &__count_to_two, size_t &__range_length,`
`const bool __make_twice)`
- `void __yield ()`
- `template<typename _Iter, typename _FunctorType >`
`size_t list_partition (const _Iter __begin, const _Iter __end, _Iter *__starts, size_t *__lengths, const int __`
`num_parts, _FunctorType &__f, int __oversampling=0)`
- `template<typename _Tp >`
`const _Tp &max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp >`
`const _Tp &min (const _Tp &__a, const _Tp &__b)`
- `template<typename _RanSeqs, typename _RankType, typename _RankIterator, typename _Compare >`
`void multiseq_partition (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankIterator`
`__begin_offsets, _Compare __comp=std::less< typename std::iterator_traits< typename std::iterator_traits<`
`_RanSeqs >::value_type::first_type >::value_type >())`
- `template<typename _Tp, typename _RanSeqs, typename _RankType, typename _Compare >`
`_Tp multiseq_selection (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankType &__`
`offset, _Compare __comp=std::less< _Tp >())`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut`
`__target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::exact_tag __tag)`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut`
`__target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sampling_tag __tag)`

- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut <←`
`__target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut <←`
`__target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut <←`
`__target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator, typename _RAIter3, typename <←`
`DifferenceTp, typename _Compare >`
`_RAIter3 multiway_merge_3_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 <←`
`target, _DifferenceTp __length, _Compare __comp)`
- `template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator, typename _RAIter3, typename <←`
`DifferenceTp, typename _Compare >`
`_RAIter3 multiway_merge_4_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 <←`
`target, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >`
`void multiway_merge_exact_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _Difference<←`
`Type __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, <←`
`std::pair< _DifferenceType, <←`
`std::pair< _DifferenceType, <←`
`DifferenceType > > * __pieces)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 multiway_merge_loser_tree (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 <←`
`__target, _DifferenceTp __length, _Compare __comp)`
- `template<typename _UnguardedLoserTree, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare`
`>`
`_RAIter3 multiway_merge_loser_tree_sentinel (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, <←`
`_RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value<←`
`type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 multiway_merge_loser_tree_unguarded (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, <←`
`_RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value<←`
`type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >`
`void multiway_merge_sampling_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, <←`
`DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< <←`
`std::pair< <←`
`DifferenceType, _DifferenceType > > * __pieces)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, <←`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, <←`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, <←`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, <←`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, <←`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`

- `template<bool __stable, bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Splitter, typename _Compare >`
`_RAIter3 parallel_multiway_merge (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _Splitter __splitter, _DifferenceTp __length, _Compare __comp, _ThreadIndex __num_threads)`
- `template<bool __stable, bool __exact, typename _RAIter, typename _Compare >`
`void parallel_sort_mwms (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<bool __stable, bool __exact, typename _RAIter, typename _Compare >`
`void parallel_sort_mwms_pu (_PMWMSortingData< _RAIter > *__sd, _Compare &__comp)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`

Variables

- static const int [_CASable_bits](#)
- static const [_CASable](#) [_CASable_mask](#)

4.6.1 Detailed Description

GNU parallel code for public use.

4.6.2 Typedef Documentation

`_BinIndex`

```
typedef unsigned short \_\_gnu\_parallel::\_BinIndex
```

Type to hold the index of a bin.

Since many variables of this type are allocated, it should be chosen as small as possible.

_CASable

```
typedef int64_t __gnu_parallel::_CASable
```

Longest compare-and-swappable integer type on this platform.

_SequenceIndex

```
typedef uint64_t __gnu_parallel::_SequenceIndex
```

Unsigned integer to index `__elements`. The total number of elements for each algorithm must fit into this type.

_ThreadIndex

```
typedef uint16_t __gnu_parallel::_ThreadIndex
```

Unsigned integer to index a thread number. The maximum thread number (for each processor) must fit into this type.

4.6.3 Enumeration Type Documentation**_AlgorithmStrategy**

```
enum __gnu_parallel::_AlgorithmStrategy
```

Strategies for run-time algorithm selection:

_FindAlgorithm

```
enum __gnu_parallel::_FindAlgorithm
```

Find algorithms:

_MultiwayMergeAlgorithm

```
enum __gnu_parallel::_MultiwayMergeAlgorithm
```

Merging algorithms:

_Parallelism

```
enum __gnu_parallel::_Parallelism
```

Run-time equivalents for the compile-time tags.

Enumerator

<code>sequential</code>	Not parallel.
<code>parallel_unbalanced</code>	Parallel unbalanced (equal-sized chunks).
<code>parallel_balanced</code>	Parallel balanced (work-stealing).
<code>parallel_omp_loop</code>	Parallel with OpenMP dynamic load-balancing.
<code>parallel_omp_loop_static</code>	Parallel with OpenMP static load-balancing.
<code>parallel_taskqueue</code>	Parallel with OpenMP taskqueue construct.

_PartialSumAlgorithm

```
enum __gnu_parallel::_PartialSumAlgorithm
```

Partial sum algorithms: recursive, linear.

_SortAlgorithm

```
enum __gnu_parallel::_SortAlgorithm
```

Sorting algorithms:

`__SplittingAlgorithm`

enum `__gnu_parallel::__SplittingAlgorithm`

Sorting/merging algorithms: `sampling`, `__exact`.

4.6.4 Function Documentation

`__calc_borders()`

```
template<typename _RAIter , typename _DifferenceTp >
void __gnu_parallel::__calc_borders (
    _RAIter __elements,
    _DifferenceTp __length,
    _DifferenceTp * __off )
```

Precalculate `__advances` for Knuth-Morris-Pratt algorithm.

Parameters

<code>__elements</code>	Begin iterator of sequence to search for.
<code>__length</code>	Length of sequence to search for.
<code>__off</code>	Returned <code>__offsets</code> .

Referenced by `__search_template()`.

`__compare_and_swap()`

```
template<typename _Tp >
bool __gnu_parallel::__compare_and_swap (
    volatile _Tp * __ptr,
    _Tp __comparand,
    _Tp __replacement ) [inline]
```

Compare-and-swap.

Compare `*__ptr` and `__comparand`. If equal, let `*__ptr=__replacement` and return true, return false otherwise.

Parameters

<code>__ptr</code>	Pointer to signed integer.
<code>__comparand</code>	Compare value.
<code>__replacement</code>	Replacement value.

Referenced by `__parallel_partition()`, `__gnu_parallel::__RestrictedBoundedConcurrentQueue<_Tp>::pop_back()`, and `__gnu_parallel::__RestrictedBoundedConcurrentQueue<_Tp>::pop_front()`.

`__decode2()`

```
void __gnu_parallel::__decode2 (
    _CASable __x,
    int & __a,
    int & __b ) [inline]
```

Decode two integers from one `gnu_parallel::__CASable`.

Parameters

<code>__gnu_parallel::_CASable</code> to decode integers from.	
<code>__x</code>	
<code>__a</code>	First integer, to be decoded from the most-significant <code>_CASable_bits/2</code> bits of <code>__x</code> .
<code>__b</code>	Second integer, to be encoded in the least-significant <code>_CASable_bits/2</code> bits of <code>__x</code> .

See also

[`__encode2`](#)

References [`_CASable_bits`](#), and [`_CASable_mask`](#).

Referenced by [`__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::pop_back\(\)`](#), [`__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::push_front\(\)`](#), and [`__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::push_front\(\)`](#).

[`__determine_samples\(\)`](#)

```
template<typename _RAIter , typename _DifferenceTp >
void __gnu_parallel::__determine_samples (
    \_PMWMSSortingData<\_RAIter>\* __sd,
    _DifferenceTp __num_samples )
```

Select `_M_samples` from a sequence.

Parameters

<code>__sd</code>	Pointer to algorithm data. Result will be placed in <code>__sd->_M_samples</code> .
<code>__num_samples</code>	Number of <code>_M_samples</code> to select.

References [`__equally_split\(\)`](#), [`__gnu_parallel::_PMWMSSortingData<_RAIter>::_M_samples`](#), [`__gnu_parallel::_PMWMSSortingData<_RAIter>::_M_starts`](#), and [`__gnu_parallel::_PMWMSSortingData<_RAIter>::_M_starts`](#).

[`__encode2\(\)`](#)

```
\_CASable __gnu_parallel::__encode2 (
    int __a,
    int __b ) [inline]
```

Encode two integers into one `gnu_parallel::_CASable`.

Parameters

<code>__a</code>	First integer, to be encoded in the most-significant <code>_CASable_bits/2</code> bits.
<code>__b</code>	Second integer, to be encoded in the least-significant <code>_CASable_bits/2</code> bits.

Returns

value encoding `__a` and `__b`.

See also

`__decode2`

References [_CASable_bits](#).

Referenced by [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::_RestrictedBoundedConcurrentQueue\(\)](#), [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::pop_back\(\)](#), [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::push_front\(\)](#), and [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::push_front\(\)](#).

`__equally_split()`

```
template<typename _DifferenceType , typename _OutputIterator >
_OutputIterator __gnu_parallel::__equally_split (
    _DifferenceType __n,
    _ThreadIndex __num_threads,
    _OutputIterator __s )
```

function to split a sequence into parts of almost equal size.

The resulting sequence `__s` of length `__num_threads+1` contains the splitting positions when splitting the range `[0,__n)` into parts of almost equal size (plus minus 1). The first entry is 0, the last one `n`. There may result empty parts.

Parameters

<code>__n</code>	Number of elements
<code>__num_threads</code>	Number of parts
<code>__s</code>	Splitters

Returns

End of `__splitter` sequence, i.e. `__s+__num_threads+1`

Referenced by [__determine_samples\(\)](#), [__find_template\(\)](#), [__parallel_partial_sum_linear\(\)](#), [__parallel_unique_copy\(\)](#), [__search_template\(\)](#), and [multiway_merge_exact_splitting\(\)](#).

`__equally_split_point()`

```
template<typename _DifferenceType >
_DifferenceType __gnu_parallel::__equally_split_point (
    _DifferenceType __n,
    _ThreadIndex __num_threads,
    _ThreadIndex __thread_no )
```

function to split a sequence into parts of almost equal size.

Returns the position of the splitting point between thread number `__thread_no` (included) and thread number `__thread_no+1` (excluded).

Parameters

<code>__n</code>	Number of elements
<code>__num_threads</code>	Number of parts
<code>__thread_no</code>	Number of threads

Returns

splitting point

Referenced by [__for_each_template_random_access_ed\(\)](#).

__fetch_and_add()

```
template<typename _Tp >
_Tp __gnu_parallel::__fetch_and_add (
    volatile _Tp * __ptr,
    _Tp __addend ) [inline]
```

Add a value to a variable, atomically.

Parameters

<code>__ptr</code>	Pointer to a signed integer.
<code>__addend</code>	Value to add.

Referenced by [__parallel_partition\(\)](#), and [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>::push_front\(\)](#).

__find_template() [1/4]

```
template<typename _RAIter1 , typename _RAIter2 , typename _Pred , typename _Selector >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred,
    _Selector __selector ) [inline]
```

Parallel `std::find`, switch for different algorithms.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	_Functionality (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

References [__find_template\(\)](#), and [__gnu_parallel::_Settings::get\(\)](#).

Referenced by [__find_template\(\)](#).

__find_template() [2/4]

```
template<typename _RAIter1 , typename _RAIter2 , typename _Pred , typename _Selector >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred,
    _Selector __selector,
    constant_size_blocks_tag )
```

Parallel `std::find`, constant block size variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second <code>__sequence</code> must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	<code>_Functionality</code> (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

See also

`__gnu_parallel::Settings::find_sequential_search_size`

`__gnu_parallel::Settings::find_block_size` There are two main differences between the growing blocks and the constant-size blocks variants.

1. For GB, the block size grows; for CSB, the block size is fixed.
2. For GB, the blocks are allocated dynamically; for CSB, the blocks are allocated in a predetermined manner, namely spacial round-robin.

References [GLIBCXX_CALL](#), `__gnu_parallel::Settings::find_initial_block_size`, `__gnu_parallel::Settings::find_sequential_search_size`, `std::pair<_T1, _T2>::first`, and `__gnu_parallel::Settings::get()`.

`__find_template()` [3/4]

```
template<typename _RAIter1 , typename _RAIter2 , typename _Pred , typename _Selector >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred,
    _Selector __selector,
    equal_split_tag )
```

Parallel `std::find`, equal splitting variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second <code>__sequence</code> must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	<code>_Functionality</code> (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

References [__equally_split\(\)](#), and [_GLIBCXX_CALL](#).

`__find_template()` [4/4]

```
template<typename _RAIter1 , typename _RAIter2 , typename _Pred , typename _Selector >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred,
    _Selector __selector,
    growing_blocks_tag )
```

Parallel `std::find`, growing block size variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second <code>__sequence</code> must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	<code>_Functionality</code> (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

See also

`__gnu_parallel::Settings::find_sequential_search_size`

`__gnu_parallel::Settings::find_scale_factor`

There are two main differences between the growing blocks and the constant-size blocks variants.

1. For GB, the block size grows; for CSB, the block size is fixed.
2. For GB, the blocks are allocated dynamically; for CSB, the blocks are allocated in a predetermined manner, namely spacial round-robin.

References [_GLIBCXX_CALL](#), [__gnu_parallel::Settings::find_scale_factor](#), [__gnu_parallel::Settings::find_sequential_search_size](#), [std::pair<_T1, _T2>::first](#), and [__gnu_parallel::Settings::get\(\)](#).

`__for_each_template_random_access()`

```
template<typename _IIter , typename _UserOp , typename _Functionality , typename _Red , typename
_Result >
_UserOp __gnu_parallel::__for_each_template_random_access (
    _IIter __begin,
    _IIter __end,
    _UserOp __user_op,
    _Functionality & __functionality,
    _Red __reduction,
```

```

_Result __reduction_start,
_Result & __output,
typename std::iterator_traits< _IIter >::difference_type __bound,
_Parallelism __parallelism_tag )

```

Chose the desired algorithm by evaluating `__parallelism_tag`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__user_op</code>	A user-specified functor (comparator, predicate, associative operator,...)
<code>__functionality</code>	functor to <i>process</i> an element with <code>__user_op</code> (depends on desired functionality, e. g. accumulate, <code>for_each</code> ,...)
<code>__reduction</code>	Reduction functor.
<code>__reduction_start</code>	Initial value for reduction.
<code>__output</code>	Output iterator.
<code>__bound</code>	Maximum number of elements processed.
<code>__parallelism_tag</code>	Parallelization method

References [__for_each_template_random_access_ed\(\)](#), [__for_each_template_random_access_omp_loop\(\)](#), [__for_each_template_random_access_omp_loop_static\(\)](#), [parallel_omp_loop](#), [parallel_omp_loop_static](#), and [parallel_unbalanced](#).

`__for_each_template_random_access_ed()`

```

template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_ed (
    _RAIter __begin,
    _RAIter __end,
    _Op __o,
    _Fu & __f,
    _Red __r,
    _Result __base,
    _Result & __output,
    typename std::iterator_traits< _RAIter >::difference_type __bound )

```

Embarrassingly parallel algorithm for random access iterators, using hand-crafted parallelization by equal splitting the work.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, ...)
<code>__f</code>	Functor to “process” an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to “add” a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

References [__equally_split_point\(\)](#).

Referenced by [__for_each_template_random_access\(\)](#).

`__for_each_template_random_access_omp_loop()`

```
template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_omp_loop (
    _RAIter __begin,
    _RAIter __end,
    _Op __o,
    _Fu & __f,
    _Red __r,
    _Result __base,
    _Result & __output,
    typename std::iterator_traits< _RAIter >::difference_type __bound )
```

Embarrassingly parallel algorithm for random access iterators, using an OpenMP for loop.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, etc.).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

Referenced by [__for_each_template_random_access\(\)](#).

`__for_each_template_random_access_omp_loop_static()`

```
template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_omp_loop_static (
    _RAIter __begin,
    _RAIter __end,
    _Op __o,
    _Fu & __f,
    _Red __r,
    _Result __base,
    _Result & __output,
    typename std::iterator_traits< _RAIter >::difference_type __bound )
```

Embarrassingly parallel algorithm for random access iterators, using an OpenMP for loop with static scheduling.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
----------------------	-------------------------------------

Parameters

<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, ...).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed <code>__elements</code> (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

`__for_each_template_random_access_workstealing()`

```
template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_workstealing (
    _RAIter __begin,
    _RAIter __end,
    _Op __op,
    _Fu & __f,
    _Red __r,
    _Result __base,
    _Result & __output,
    typename std::iterator_traits< _RAIter >::difference_type __bound )
```

Work stealing algorithm for random access iterators.

Uses $O(1)$ additional memory. Synchronization at job lists is done with atomic operations.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__op</code>	User-supplied functor (comparator, predicate, adding functor, ...).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

References [__yield\(\)](#), [_GLIBCXX_CALL](#), [__gnu_parallel::__Job<_DifferenceTp>::__M_first](#), [__gnu_parallel::__Job<_DifferenceTp>::__M_load](#), [__gnu_parallel::__Settings::cache_line_size](#), [__gnu_parallel::__Settings::get\(\)](#), and [min\(\)](#).

Referenced by [__for_each_template_random_access\(\)](#).

__is_sorted()

```
template<typename _IIter , typename _Compare >
bool __gnu_parallel::__is_sorted (
    _IIter __begin,
    _IIter __end,
    _Compare __comp )
```

Check whether [[__begin](#), [__end](#)) is sorted according to [__comp](#).

Parameters

__begin	Begin iterator of sequence.
__end	End iterator of sequence.
__comp	Comparator.

Returns

`true` if sorted, `false` otherwise.

Referenced by [__sequential_multiway_merge\(\)](#), [multiway_merge_loser_tree_sentinel\(\)](#), and [parallel_multiway_merge\(\)](#).

__median_of_three_iterators()

```
template<typename _RAIter , typename _Compare >
_RAIter __gnu_parallel::__median_of_three_iterators (
    _RAIter __a,
    _RAIter __b,
    _RAIter __c,
    _Compare __comp )
```

Compute the median of three referenced elements, according to [__comp](#).

Parameters

__a	First iterator.
__b	Second iterator.
__c	Third iterator.
__comp	Comparator.

Referenced by [__qsb_divide\(\)](#).

__merge_advance()

```
template<typename _RAIter1 , typename _RAIter2 , typename _OutputIterator , typename _Difference←
Tp , typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance (
```

```

    _RAIter1 & __begin1,
    _RAIter1 __end1,
    _RAIter2 & __begin2,
    _RAIter2 __end2,
    _OutputIterator __target,
    _DifferenceTp __max_length,
    _Compare __comp ) [inline]

```

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. Static switch on whether to use the conditional-move variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

References [__merge_advance_movc\(\)](#), and [_GLIBCXX_CALL](#).

Referenced by [__parallel_merge_advance\(\)](#), and [__sequential_multiway_merge\(\)](#).

`__merge_advance_movc()`

```

template<typename _RAIter1 , typename _RAIter2 , typename _OutputIterator , typename _Difference←
Tp , typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_movc (
    _RAIter1 & __begin1,
    _RAIter1 __end1,
    _RAIter2 & __begin2,
    _RAIter2 __end2,
    _OutputIterator __target,
    _DifferenceTp __max_length,
    _Compare __comp )

```

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. Specially designed code should allow the compiler to generate conditional moves instead of branches.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Referenced by [__merge_advance\(\)](#).

[__merge_advance_usual\(\)](#)

```
template<typename _RAIter1 , typename _RAIter2 , typename _OutputIterator , typename _DifferenceType,
        typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_usual (
    _RAIter1 & __begin1,
    _RAIter1 __end1,
    _RAIter2 & __begin2,
    _RAIter2 __end2,
    _OutputIterator __target,
    _DifferenceType __max_length,
    _Compare __comp )
```

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

[__parallel_merge_advance\(\)](#) [1/2]

```
template<typename _RAIter1 , typename _RAIter3 , typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (
    _RAIter1 & __begin1,
    _RAIter1 __end1,
    _RAIter1 & __begin2,
    _RAIter1 __end2,
    _RAIter3 __target,
    typename std::iterator_traits< _RAIter1 >::difference_type __max_length,
    _Compare __comp ) [inline]
```

Parallel merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. The functionality is projected onto `parallel_multiway_merge`.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
-----------------------	-----------------------------------

Parameters

<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

References [multiway_merge_exact_splitting\(\)](#), and [parallel_multiway_merge\(\)](#).

`__parallel_merge_advance()` [2/2]

```
template<typename _RAIter1 , typename _RAIter2 , typename _RAIter3 , typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (
    _RAIter1 & __begin1,
    _RAIter1 __end1,
    _RAIter2 & __begin2,
    _RAIter2 __end2,
    _RAIter3 __target,
    typename std::iterator_traits< _RAIter1 >::difference_type __max_length,
    _Compare __comp ) [inline]
```

Merge routine fallback to sequential in case the iterators of the two input sequences are of different type.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

References [__merge_advance\(\)](#).

`__parallel_nth_element()`

```
template<typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_nth_element (
    _RAIter __begin,
    _RAIter __nth,
    _RAIter __end,
    _Compare __comp )
```

Parallel implementation of `std::nth_element()`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__nth</code>	Iterator of element that must be in position afterwards.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

References [__parallel_partition\(\)](#), [_GLIBCXX_CALL](#), [__gnu_parallel::_Settings::get\(\)](#), [std::max\(\)](#), [__gnu_parallel::_Settings::nth_element](#) and [__gnu_parallel::_Settings::partition_minimal_n](#).

Referenced by [__parallel_partial_sort\(\)](#).

`__parallel_partial_sort()`

```
template<typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_partial_sort (
    _RAIter __begin,
    _RAIter __middle,
    _RAIter __end,
    _Compare __comp )
```

Parallel implementation of `std::partial_sort()`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__middle</code>	Sort until this position.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

References [__parallel_nth_element\(\)](#).

`__parallel_partial_sum()`

```
template<typename _IIter , typename _OutputIterator , typename _BinaryOperation >
_OutputIterator __gnu_parallel::__parallel_partial_sum (
    _IIter __begin,
    _IIter __end,
    _OutputIterator __result,
    _BinaryOperation __bin_op )
```

Parallel partial sum front-`__end`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.

Returns

End iterator of output sequence.

References [__parallel_partial_sum_linear\(\)](#), [_GLIBCXX_CALL](#), and [__gnu_parallel::_Settings::get\(\)](#).

__parallel_partial_sum_basecase()

```
template<typename _IIter , typename _OutputIterator , typename _BinaryOperation >
_OutputIterator __gnu_parallel::__parallel_partial_sum_basecase (
    _IIter __begin,
    _IIter __end,
    _OutputIterator __result,
    _BinaryOperation __bin_op,
    typename std::iterator_traits< _IIter >::value_type __value )
```

Base case prefix sum routine.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.
<code>__value</code>	Start value. Must be passed since the neutral element is unknown in general.

Returns

End iterator of output sequence.

Referenced by [__parallel_partial_sum_linear\(\)](#).

__parallel_partial_sum_linear()

```
template<typename _IIter , typename _OutputIterator , typename _BinaryOperation >
_OutputIterator __gnu_parallel::__parallel_partial_sum_linear (
    _IIter __begin,
    _IIter __end,
    _OutputIterator __result,
    _BinaryOperation __bin_op,
    typename std::iterator_traits< _IIter >::difference_type __n )
```

Parallel partial sum implementation, two-phase approach, no recursion.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.
<code>__n</code>	Length of sequence.

Returns

End iterator of output sequence.

References [__equally_split\(\)](#), [__parallel_partial_sum_basecase\(\)](#), [__gnu_parallel::_Settings::get\(\)](#), and [__gnu_parallel::_Settings::partial](#)

Referenced by [__parallel_partial_sum\(\)](#).

__parallel_partition()

```
template<typename _RAIter , typename _Predicate >
```

```
std::iterator_traits<_RAIter >::difference_type __gnu_parallel::__parallel_partition (
    _RAIter __begin,
    _RAIter __end,
    _Predicate __pred,
    _ThreadIndex __num_threads )
```

Parallel implementation of `std::partition`.

Parameters

<code>__begin</code>	Begin iterator of input sequence to split.
<code>__end</code>	End iterator of input sequence to split.
<code>__pred</code>	Partition predicate, possibly including some kind of pivot.
<code>__num_threads</code>	Maximum number of threads to use for this task.

Returns

Number of elements not fulfilling the predicate.

References [__compare_and_swap\(\)](#), [__fetch_and_add\(\)](#), [_GLIBCXX_CALL](#), [_GLIBCXX_VOLATILE](#), [__gnu_parallel::_Settings::get\(\)](#), [__gnu_parallel::_Settings::partition_chunk_share](#), and [__gnu_parallel::_Settings::partition_chunk_size](#).

Referenced by [__parallel_nth_element\(\)](#), [__parallel_sort_qs_divide\(\)](#), and [__qsb_divide\(\)](#).

`__parallel_random_shuffle()`

```
template<typename _RAIter , typename _RandomNumberGenerator >
void __gnu_parallel::__parallel_random_shuffle (
    _RAIter __begin,
    _RAIter __end,
    _RandomNumberGenerator __rng = _RandomNumber() ) [inline]
```

Parallel random public call.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__rng</code>	Random number generator to use.

References [__parallel_random_shuffle_drs\(\)](#).

`__parallel_random_shuffle_drs()`

```
template<typename _RAIter , typename _RandomNumberGenerator >
void __gnu_parallel::__parallel_random_shuffle_drs (
    _RAIter __begin,
    _RAIter __end,
    typename std::iterator_traits<_RAIter >::difference_type __n,
    _ThreadIndex __num_threads,
    _RandomNumberGenerator & __rng )
```

Main parallel random shuffle step.

Parameters

<code>__begin</code>	Begin iterator of sequence.
----------------------	-----------------------------

Parameters

<code>__end</code>	End iterator of sequence.
<code>__n</code>	Length of sequence.
<code>__num_threads</code>	Number of threads to use.
<code>__rng</code>	Random number generator to use.

References [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__bins_end](#), [__parallel_random_shuffle_drs_pu\(\)](#), [__rd_log2\(\)](#), [__round_up_to_pow2\(\)](#), [__sequential_random_shuffle\(\)](#), [_GLIBCXX_CALL](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_bin_dist](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_dist](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_dist](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_sd](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_source](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_source](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_temporaries](#), [__gnu_parallel::Settings::get\(\)](#), [__gnu_parallel::Settings::L2_cache_size](#), [std::min\(\)](#), and [__gnu_parallel::Settings::TLB_size](#).

Referenced by [__parallel_random_shuffle\(\)](#).

`__parallel_random_shuffle_drs_pu()`

```
template<typename _RAIter, typename _RandomNumberGenerator>
void __gnu_parallel::__parallel_random_shuffle_drs_pu (
    __DRSSorterPU<_RAIter, _RandomNumberGenerator> * __pus )
```

Random shuffle code executed by each thread.

Parameters

<code>__pus</code>	Array of thread-local data records.
--------------------	-------------------------------------

References [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__bins_end](#), [__random_number_pow2\(\)](#), [__sequential_random_shuffle\(\)](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_bin_dist](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_dist](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_dist](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_sd](#), [__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_source](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_source](#), [__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_temporaries](#), and [std::partial_sum\(\)](#).

Referenced by [__parallel_random_shuffle_drs\(\)](#).

`__parallel_sort()` [1/7]

```
template<bool __stable, typename _RAIter, typename _Compare>
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    balanced_quicksort_tag __parallelism ) [inline]
```

Choose balanced quicksort for parallel sorting.

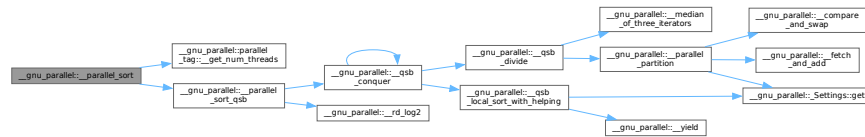
Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), [__parallel_sort_qsb\(\)](#), and [_GLIBCXX_CALL](#).
Here is the call graph for this function:

**`__parallel_sort()`** [2/7]

```
template<bool __stable, typename _RAIter, typename _Compare>
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    default_parallel_tag __parallelism ) [inline]
```

Choose multiway mergesort with exact splitting, for parallel sorting.

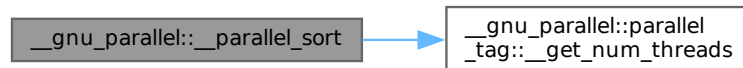
Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), and [_GLIBCXX_CALL](#).
Here is the call graph for this function:



__parallel_sort() [3/7]

```
template<bool __stable, typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    multiway_mergesort_exact_tag __parallelism ) [inline]
```

Choose multiway mergesort with exact splitting, for parallel sorting.

Parameters

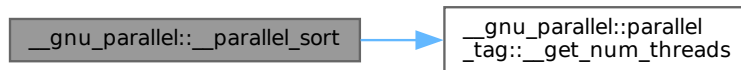
<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), and [_GLIBCXX_CALL](#).

Here is the call graph for this function:

**__parallel_sort()** [4/7]

```
template<bool __stable, typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    multiway_mergesort_sampling_tag __parallelism ) [inline]
```

Choose multiway mergesort with splitting by sampling, for parallel sorting.

Parameters

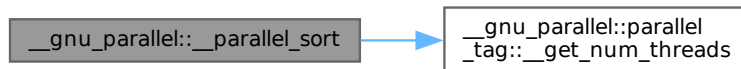
<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), and [_GLIBCXX_CALL](#).

Here is the call graph for this function:

**`__parallel_sort()`** [5/7]

```

template<bool __stable, typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    multiway_mergesort_tag __parallelism ) [inline]
  
```

Choose multiway mergesort, splitting variant at run-time, for parallel sorting.

Parameters

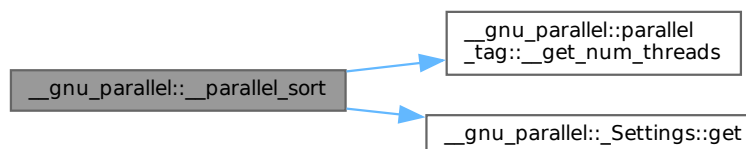
<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), [_GLIBCXX_CALL](#), and [__gnu_parallel::_Settings::get\(\)](#).

Here is the call graph for this function:



__parallel_sort() [6/7]

```
template<bool __stable, typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    parallel_tag __parallelism ) [inline]
```

Choose a parallel sorting algorithm.

Parameters

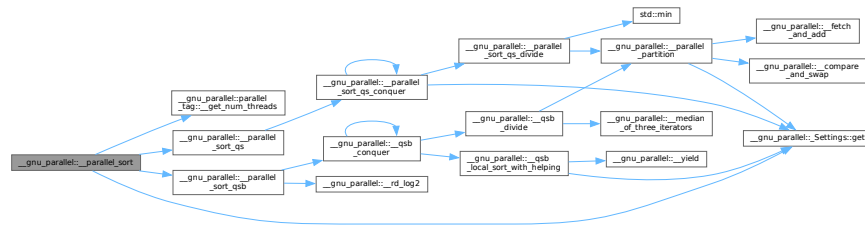
<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), [__parallel_sort_qs\(\)](#), [__parallel_sort_qsb\(\)](#), [_GLIBCXX_CALL](#), and [__gnu_parallel::Settings::get\(\)](#).

Here is the call graph for this function:

**__parallel_sort()** [7/7]

```
template<bool __stable, typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    quicksort_tag __parallelism ) [inline]
```

Choose quicksort for parallel sorting.

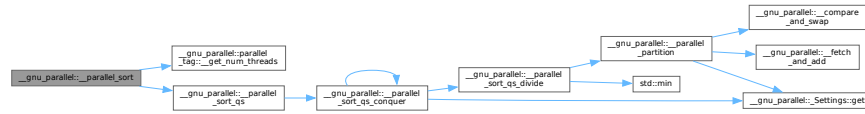
Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

References [__gnu_parallel::parallel_tag::__get_num_threads\(\)](#), [__parallel_sort_qs\(\)](#), and [_GLIBCXX_CALL](#).
Here is the call graph for this function:

`__parallel_sort_qs()`

```

template<typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort_qs (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __num_threads )

```

Unbalanced quicksort main call.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator input sequence, ignored.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

References [__parallel_sort_qs_conquer\(\)](#), and [_GLIBCXX_CALL](#).
Referenced by [__parallel_sort\(\)](#), and [__parallel_sort\(\)](#).

`__parallel_sort_qs_conquer()`

```

template<typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort_qs_conquer (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __num_threads )

```

Unbalanced quicksort conquer step.

Parameters

<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

References [__parallel_sort_qs_conquer\(\)](#), [__parallel_sort_qs_divide\(\)](#), and [__gnu_parallel::_Settings::get\(\)](#).
 Referenced by [__parallel_sort_qs\(\)](#), and [__parallel_sort_qs_conquer\(\)](#).

`__parallel_sort_qs_divide()`

```
template<typename _RAIter , typename _Compare >
std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__parallel_sort_qs_divide (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    typename std::iterator_traits< _RAIter >::difference_type __pivot_rank,
    typename std::iterator_traits< _RAIter >::difference_type __num_samples,
    _ThreadIndex __num_threads )
```

Unbalanced quicksort divide step.

Parameters

<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__pivot_rank</code>	Desired <code>__rank</code> of the pivot.
<code>__num_samples</code>	Choose pivot from that many samples.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

References [__parallel_partition\(\)](#), and [std::min\(\)](#).
 Referenced by [__parallel_sort_qs_conquer\(\)](#).

`__parallel_sort_qsb()`

```
template<typename _RAIter , typename _Compare >
void __gnu_parallel::__parallel_sort_qsb (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __num_threads )
```

Top-level quicksort routine.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

References [__qsb_conquer\(\)](#), [__rd_log2\(\)](#), [_GLIBCXX_CALL](#), and [__gnu_parallel::_QSBThreadLocal< _RAIter >::_M_elements_leftover](#).
 Referenced by [__parallel_sort\(\)](#), and [__parallel_sort\(\)](#).

`__parallel_unique_copy()` [1/2]

```
template<typename _IIter , class _OutputIterator >
_OutputIterator __gnu_parallel::__parallel_unique_copy (
    _IIter __first,
```

```

_Iter __last,
_OutputIterator __result ) [inline]

```

Parallel `std::unique_copy()`, without explicit equality predicate.

Parameters

<code>__first</code>	Begin iterator of input sequence.
<code>__last</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of result <code>__sequence</code> .

Returns

End iterator of result `__sequence`.

References [__parallel_unique_copy\(\)](#).

`__parallel_unique_copy()` [2/2]

```

template<typename _IIter , class _OutputIterator , class _BinaryPredicate >
_OutputIterator __gnu_parallel::__parallel_unique_copy (
    _IIter __first,
    _IIter __last,
    _OutputIterator __result,
    _BinaryPredicate __binary_pred )

```

Parallel `std::unique_copy()`, w/`__o` explicit equality predicate.

Parameters

<code>__first</code>	Begin iterator of input sequence.
<code>__last</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of result <code>__sequence</code> .
<code>__binary_pred</code>	Equality predicate.

Returns

End iterator of result `__sequence`.

References [__equally_split\(\)](#), and [_GLIBCXX_CALL](#).

Referenced by [__parallel_unique_copy\(\)](#).

`__qsb_conquer()`

```

template<typename _RAIter , typename _Compare >
void __gnu_parallel::__qsb_conquer (
    _QSBThreadLocal< _RAIter > ** __tls,
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __iam,
    _ThreadIndex __num_threads,
    bool __parent_wait )

```

Quicksort conquer step.

Parameters

<code>__tls</code>	Array of thread-local storages.
<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__iam</code>	Number of the thread processing this function.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

References [__qsb_conquer\(\)](#), [__qsb_divide\(\)](#), [__qsb_local_sort_with_helping\(\)](#), [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_element\(\)](#), and [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_initial\(\)](#).

Referenced by [__parallel_sort_qsb\(\)](#), and [__qsb_conquer\(\)](#).

`__qsb_divide()`

```
template<typename _RAIter , typename _Compare >
std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__qsb_divide (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __num_threads )
```

Balanced quicksort divide step.

Parameters

<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

Precondition

`(__end-__begin)>=1`

References [__median_of_three_iterators\(\)](#), and [__parallel_partition\(\)](#).

Referenced by [__qsb_conquer\(\)](#).

`__qsb_local_sort_with_helping()`

```
template<typename _RAIter , typename _Compare >
void __gnu_parallel::__qsb_local_sort_with_helping (
    _QSBThreadLocal< _RAIter > ** __tls,
    _Compare & __comp,
    _ThreadIndex __iam,
    bool __wait )
```

Quicksort step doing load-balanced local sort.

Parameters

<code>__tls</code>	Array of thread-local storages.
<code>__comp</code>	Comparator.
<code>__iam</code>	Number of the thread processing this function.

References [__yield\(\)](#), [_GLIBCXX_PARALLEL_ASSERTIONS](#), [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_elements_leftover](#), [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_initial](#), [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_leftover_parts](#), [__gnu_parallel::__QSBThreadLocal<_RAIter>::__M_num_threads](#), [__gnu_parallel::_Settings::get\(\)](#), and [__gnu_parallel::_Settings::sort\(\)](#).
 Referenced by [__qsb_conquer\(\)](#).

`__random_number_pow2()`

```
template<typename _RandomNumberGenerator >
int __gnu_parallel::__random_number_pow2 (
    int __logp,
    _RandomNumberGenerator & __rng ) [inline]
```

Generate a random number in $[0, 2^{\text{__logp}})$.

Parameters

<code>__logp</code>	Logarithm (basis 2) of the upper range <code>__bound</code> .
<code>__rng</code>	Random number generator to use.

Referenced by [__parallel_random_shuffle_drs_pu\(\)](#), and [__sequential_random_shuffle\(\)](#).

`__rd_log2()`

```
template<typename _Size >
_Size __gnu_parallel::__rd_log2 (
    _Size __n ) [inline]
```

Calculates the rounded-down logarithm of `__n` for base 2.

Parameters

<code>__n</code>	Argument.
------------------	-----------

Returns

Returns 0 for any argument < 1 .

Referenced by [__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_LoserTreeBase\(\)](#), [__parallel_random_shuffle_drs\(\)](#), [__parallel_sort_qsb\(\)](#), [__round_up_to_pow2\(\)](#), [__sequential_random_shuffle\(\)](#), [multiseq_partition\(\)](#), and [multiseq_selection\(\)](#).

`__round_up_to_pow2()`

```
template<typename _Tp >
_Tp __gnu_parallel::__round_up_to_pow2 (
    _Tp __x )
```

Round up to the next greater power of 2.

Parameters

<code>__x</code>	Integer to round up
------------------	---------------------

References [__rd_log2\(\)](#).

Referenced by [__parallel_random_shuffle_drs\(\)](#), [__sequential_random_shuffle\(\)](#), and [multiseq_selection\(\)](#).

__search_template()

```
template<typename __RAIter1 , typename __RAIter2 , typename _Pred >
__RAIter1 __gnu_parallel::__search_template (
    __RAIter1 __begin1,
    __RAIter1 __end1,
    __RAIter2 __begin2,
    __RAIter2 __end2,
    _Pred __pred )
```

Parallel `std::search`.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__pred</code>	Find predicate.

Returns

Place of finding in first sequences.

References [__calc_borders\(\)](#), [__equally_split\(\)](#), [_GLIBCXX_CALL](#), and [std::min\(\)](#).

__sequential_multiway_merge()

```
template<bool __stable, bool __sentinels, typename _RAIterIterator , typename _RAIter3 , typename
_DifferenceTp , typename _Compare >
_RAIter3 __gnu_parallel::__sequential_multiway_merge (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    const typename std::iterator\_traits< typename std::iterator\_traits< _RAIterIterator
>::value_type::first_type >::value_type & __sentinel,
    _DifferenceTp __length,
    _Compare __comp )
```

Sequential multi-way merging switch.

The `_GLIBCXX_PARALLEL_DECISION` is based on the branching factor and runtime settings.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.
<code>__sentinel</code>	The sequences have <code>__a</code> sentinel element.

Returns

End iterator of output sequence.

References [__is_sorted\(\)](#), [__merge_advance\(\)](#), [_GLIBCXX_CALL](#), and [_GLIBCXX_PARALLEL_LENGTH](#).
Referenced by [multiway_merge\(\)](#), and [multiway_merge_sentinels\(\)](#).

`__sequential_random_shuffle()`

```
template<typename _RAIter, typename _RandomNumberGenerator >
void __gnu_parallel::__sequential_random_shuffle (
    _RAIter __begin,
    _RAIter __end,
    _RandomNumberGenerator & __rng )
```

Sequential cache-efficient random shuffle.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__rng</code>	Random number generator to use.

References [__random_number_pow2\(\)](#), [__rd_log2\(\)](#), [__round_up_to_pow2\(\)](#), [__sequential_random_shuffle\(\)](#), [__gnu_parallel::Settings::get\(\)](#), [__gnu_parallel::Settings::L2_cache_size](#), [std::min\(\)](#), [std::partial_sum\(\)](#), and [__gnu_parallel::Settings::TLB_size](#).
Referenced by [__parallel_random_shuffle_drs\(\)](#), [__parallel_random_shuffle_drs_pu\(\)](#), and [__sequential_random_shuffle\(\)](#).

`__shrink()`

```
template<typename _IIter >
void __gnu_parallel::__shrink (
    std::vector<_IIter > & __os_starts,
    size_t & __count_to_two,
    size_t & __range_length )
```

Combines two ranges into one and thus halves the number of ranges.

Parameters

<code>__os_starts</code>	Start positions worked on (oversampled).
<code>__count_to_two</code>	Counts up to 2.
<code>__range_length</code>	Current length of a chunk.

References [std::vector<_Tp, _Alloc >::size\(\)](#).
Referenced by [__shrink_and_double\(\)](#).

`__shrink_and_double()`

```
template<typename _IIter >
void __gnu_parallel::__shrink_and_double (
    std::vector<_IIter > & __os_starts,
    size_t & __count_to_two,
    size_t & __range_length,
    const bool __make_twice )
```

Shrinks and doubles the ranges.

Parameters

<code>__os_starts</code>	Start positions worked on (oversampled).
<code>__count_to_two</code>	Counts up to 2.
<code>__range_length</code>	Current length of a chunk.
<code>__make_twice</code>	Whether the <code>__os_starts</code> is allowed to be grown or not

References [`__shrink\(\)`](#), [`std::vector<_Tp, _Alloc>::resize\(\)`](#), and [`std::vector<_Tp, _Alloc>::size\(\)`](#).
Referenced by [`list_partition\(\)`](#).

`__yield()`

```
void __gnu_parallel::__yield ( ) [inline]
```

Yield control to another thread, without waiting for the end of the time slice.

Referenced by [`__for_each_template_random_access_workstealing\(\)`](#), and [`__qsb_local_sort_with_helping\(\)`](#).

`list_partition()`

```
template<typename _IIter , typename _FunctorType >
size_t __gnu_parallel::list_partition (
    const _IIter __begin,
    const _IIter __end,
    _IIter * __starts,
    size_t * __lengths,
    const int __num_parts,
    _FunctorType & __f,
    int __oversampling = 0 )
```

Splits a sequence given by input iterators into parts of almost equal size.

The function needs only one pass over the sequence.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__starts</code>	Start iterators for the resulting parts, dimension <code>__num_parts+1</code> . For convenience, <code>__starts [__num_parts]</code> contains the end iterator of the sequence.
<code>__lengths</code>	Length of the resulting parts.
<code>__num_parts</code>	Number of parts to split the sequence into.
<code>__f</code>	Functor to be applied to each element by traversing <code>__it</code>
<code>__oversampling</code>	Oversampling factor. If 0, then the partitions will differ in at most $\sqrt{\text{end} - \text{begin}}$ elements. Otherwise, the ratio between the longest and the shortest part is bounded by $1/(\text{oversampling} \cdot \text{num_parts})$

Returns

Length of the whole sequence.

References [`__shrink_and_double\(\)`](#), and [`std::vector<_Tp, _Alloc>::size\(\)`](#).

`max()`

```
template<typename _Tp >
const _Tp & __gnu_parallel::max (
```

```
const _Tp & __a,
const _Tp & __b ) [inline]
```

Equivalent to `std::max`.

`min()`

```
template<typename _Tp >
const _Tp & __gnu_parallel::min (
    const _Tp & __a,
    const _Tp & __b ) [inline]
```

Equivalent to `std::min`.

Referenced by [__for_each_template_random_access_workstealing\(\)](#).

`multiseq_partition()`

```
template<typename _RanSeqs , typename _RankType , typename _RankIterator , typename _Compare >
void __gnu_parallel::multiseq_partition (
    _RanSeqs __begin_seqs,
    _RanSeqs __end_seqs,
    _RankType __rank,
    _RankIterator __begin_offsets,
    _Compare __comp = std::less< typename std::iterator\\_traits<typename std::iterator\\\_traits<\\\_RanSeqs>::value\\\_type::first\\\_type>::value\\\_type>\\\(\\\) )
```

Splits several sorted sequences at a certain global `__rank`, resulting in a splitting point for each sequence. The sequences are passed via a sequence of random-access iterator pairs, none of the sequences may be empty. If there are several equal elements across the split, the ones on the `__left` side will be chosen from sequences with smaller number.

Parameters

<code>__begin_seqs</code>	Begin of the sequence of iterator pairs.
<code>__end_seqs</code>	End of the sequence of iterator pairs.
<code>__rank</code>	The global rank to partition at.
<code>__begin_offsets</code>	A random-access <code>__sequence</code> <code>__begin</code> where the <code>__result</code> will be stored in. Each element of the sequence is an iterator that points to the first element on the greater part of the respective <code>__sequence</code> .
<code>__comp</code>	The ordering functor, defaults to <code>std::less<_Tp></code> .

References [__rd_log2\(\)](#), [_GLIBCXX_CALL](#), [std::distance\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare >::empty\(\)](#), [std::max\(\)](#), [std::min\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare >::pop\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare >::top\(\)](#) and [std::priority_queue<_Tp, _Sequence, _Compare >::top\(\)](#).

Referenced by [multiway_merge_exact_splitting\(\)](#).

`multiseq_selection()`

```
template<typename _Tp , typename _RanSeqs , typename _RankType , typename _Compare >
_Tp __gnu_parallel::multiseq_selection (
    _RanSeqs __begin_seqs,
    _RanSeqs __end_seqs,
    _RankType __rank,
    _RankType & __offset,
    _Compare __comp = std::less<\_Tp>\(\) )
```

Selects the element at a certain global `__rank` from several sorted sequences.

The sequences are passed via a sequence of random-access iterator pairs, none of the sequences may be empty.

Parameters

<code>__begin_seqs</code>	Begin of the sequence of iterator pairs.
<code>__end_seqs</code>	End of the sequence of iterator pairs.
<code>__rank</code>	The global rank to partition at.
<code>__offset</code>	The rank of the selected element in the global subsequence of elements equal to the selected element. If the selected element is unique, this number is 0.
<code>__comp</code>	The ordering functor, defaults to <code>std::less</code> .

References [__rd_log2\(\)](#), [__round_up_to_pow2\(\)](#), [GLIBCXX_CALL](#), [std::distance\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare>::std::max\(\)](#), [std::min\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare>::pop\(\)](#), [std::priority_queue<_Tp, _Sequence, _Compare>::p](#) and [std::priority_queue<_Tp, _Sequence, _Compare>::top\(\)](#).

multiway_merge()

```
template<typename _RAIterPairIterator , typename _RAIterOut , typename _DifferenceTp , typename _Compare >
_RAIterOut __gnu_parallel::multiway_merge (
    _RAIterPairIterator __seqs_begin,
    _RAIterPairIterator __seqs_end,
    _RAIterOut __target,
    _DifferenceTp __length,
    _Compare __comp,
    __gnu_parallel::sequential_tag )
```

Multiway Merge Frontend.

Merge the sequences specified by `seqs_begin` and `__seqs_end` into `__target`. `__seqs_begin` and `__seqs_end` must point to a sequence of pairs. These pairs must contain an iterator to the beginning of a sequence in their first entry and an iterator the `_M_end` of the same sequence in their second entry.

Ties are broken arbitrarily. See `stable_multiway_merge` for a variant that breaks ties by sequence number but is slower.

The first entries of the pairs (i.e. the begin iterators) will be moved forward.

The output sequence has to provide enough space for all elements that are written to it.

This function will merge the input sequences:

- not stable
- parallel, depending on the input size and Settings
- using sampling for splitting
- not using sentinels

Example:

```
int sequences[10][10];
for (int __i = 0; __i < 10; ++__i)
    for (int __j = 0; __j < 10; ++__j)
        sequences[__i][__j] = __j;

int __out[33];
std::vector<std::pair<int*> > seqs;
for (int __i = 0; __i < 10; ++__i)
    { seqs.push(std::make_pair<int*>(sequences[__i],
                                    sequences[__i] + 10)) }

multiway_merge(seqs.begin(), seqs.end(), __target, std::less<int>(), 33);
```

See also

`stable_multiway_merge`

Precondition

All input sequences must be sorted.

Target must provide enough space to merge out length elements or the number of elements in all sequences, whichever is smaller.

Postcondition

`[__target, return __value)` contains merged `__elements` from the input sequences.

`return __value - __target = min(__length, number of elements in all sequences).`

Template Parameters

<code>_RAIterPairIterator</code>	iterator over sequence of pairs of iterators
<code>_RAIterOut</code>	iterator over target sequence
<code>_DifferenceTp</code>	difference type for the sequence
<code>_Compare</code>	strict weak ordering type to compare elements in sequences

Parameters

<code>__seqs_begin</code>	<code>__begin</code> of sequence <code>__sequence</code>
<code>__seqs_end</code>	<code>_M_end</code> of sequence <code>__sequence</code>
<code>__target</code>	target sequence to merge to.
<code>__comp</code>	strict weak ordering to use for element comparison.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

`_M_end` iterator of output sequence

References [__sequential_multiway_merge\(\)](#), and [_GLIBCXX_CALL](#).

`multiway_merge_3_variant()`

```
template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator , typename
_RAIter3 , typename _DifferenceTp , typename _Compare >
_RAIter3 __gnu_parallel::multiway_merge_3_variant (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    _DifferenceTp __length,
    _Compare __comp )
```

Highly efficient 3-way merging procedure.

Merging is done with the algorithm implementation described by Peter Sanders. Basically, the idea is to minimize the number of necessary comparison after merging an element. The implementation trick that makes this fast is that the order of the sequences is stored in the instruction pointer (translated into labels in C++).

This works well for merging up to 4 sequences.

Note that making the merging stable does *not* come at a performance hit.
Whether the merging is done guarded or unguarded is selected by the used iterator class.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

References [_GLIBCXX_CALL](#).

multiway_merge_4_variant()

```
template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator , typename
_RAIter3 , typename _DifferenceTp , typename _Compare >
_RAIter3 __gnu_parallel::multiway_merge_4_variant (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    _DifferenceTp __length,
    _Compare __comp )
```

Highly efficient 4-way merging procedure.

Merging is done with the algorithm implementation described by Peter Sanders. Basically, the idea is to minimize the number of necessary comparison after merging an element. The implementation trick that makes this fast is that the order of the sequences is stored in the instruction pointer (translated into goto labels in C++).

This works well for merging up to 4 sequences.

Note that making the merging stable does *not* come at a performance hit.

Whether the merging is done guarded or unguarded is selected by the used iterator class.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

References [_GLIBCXX_CALL](#).

multiway_merge_exact_splitting()

```
template<bool __stable, typename _RAIterIterator , typename _Compare , typename _DifferenceType >
void __gnu_parallel::multiway_merge_exact_splitting (
```

```

_RAIterIterator __seqs_begin,
_RAIterIterator __seqs_end,
_DifferenceType __length,
_DifferenceType __total_length,
_Compare __comp,
std::vector< std::pair< _DifferenceType, _DifferenceType > > * __pieces )

```

Exact splitting for parallel multiway-merge routine.

None of the passed sequences may be empty.

References [__equally_split\(\)](#), [_GLIBCXX_PARALLEL_LENGTH](#), [std::vector< _Tp, _Alloc >::begin\(\)](#), [std::vector< _Tp, _Alloc >::end\(\)](#), [multiseq_partition\(\)](#), and [std::vector< _Tp, _Alloc >::resize\(\)](#).

Referenced by [__parallel_merge_advance\(\)](#).

multiway_merge_loser_tree()

```

template<typename _LT , typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp ,
typename _Compare >
_RAIter3 __gnu_parallel::multiway_merge_loser_tree (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    _DifferenceTp __length,
    _Compare __comp )

```

Multi-way merging procedure for a high branching factor, guarded case.

This merging variant uses a `LoserTree` class as selected by `_LT`.

Stability is selected through the used `LoserTree` class `_LT`.

At least one non-empty sequence is required.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

References [_GLIBCXX_CALL](#), and [_GLIBCXX_PARALLEL_LENGTH](#).

multiway_merge_loser_tree_sentinel()

```

template<typename _UnguardedLoserTree , typename _RAIterIterator , typename _RAIter3 , typename ←
_DifferenceTp , typename _Compare >
_RAIter3 __gnu_parallel::multiway_merge_loser_tree_sentinel (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator
>::value_type::first_type >::value_type & __sentinel,
    _DifferenceTp __length,
    _Compare __comp )

```

Multi-way merging procedure for a high branching factor, requiring sentinels to exist.

Template Parameters

<code>_UnguardedLoserTree</code>	Loser Tree variant to use for the unguarded merging.
----------------------------------	--

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

References [__is_sorted\(\)](#), and [_GLIBCXX_CALL](#).

`multiway_merge_loser_tree_unguarded()`

```
template<typename _LT , typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp ,
typename _Compare >
_RAIter3 __gnu_parallel::multiway_merge_loser_tree_unguarded (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator
>::value_type::first_type >::value_type & __sentinel,
    _DifferenceTp __length,
    _Compare __comp )
```

Multi-way merging procedure for a high branching factor, unguarded case.

Merging is done using the LoserTree class `_LT`.

Stability is selected by the used LoserTrees.

Precondition

No input will run out of elements during the merge.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

References [_GLIBCXX_CALL](#).

multiway_merge_sampling_splitting()

```
template<bool __stable, typename _RAIterIterator , typename _Compare , typename _DifferenceType >
void __gnu_parallel::multiway_merge_sampling_splitting (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _DifferenceType __length,
    _DifferenceType __total_length,
    _Compare __comp,
    std::vector< std::pair< _DifferenceType, _DifferenceType > > * __pieces )
```

Sampling based splitting for parallel multiway-merge routine.

References `_GLIBCXX_PARALLEL_LENGTH`, `__gnu_parallel::_Settings::get()`, and `__gnu_parallel::_Settings::merge_oversampling`.

multiway_merge_sentinels()

```
template<typename _RAIterPairIterator , typename _RAIterOut , typename _DifferenceTp , typename _Compare >
_RAIterOut __gnu_parallel::multiway_merge_sentinels (
    _RAIterPairIterator __seqs_begin,
    _RAIterPairIterator __seqs_end,
    _RAIterOut __target,
    _DifferenceTp __length,
    _Compare __comp,
    __gnu_parallel::sequential_tag )
```

Multiway Merge Frontend.

Merge the sequences specified by `seqs_begin` and `__seqs_end` into `__target`. `__seqs_begin` and `__seqs_end` must point to a sequence of pairs. These pairs must contain an iterator to the beginning of a sequence in their first entry and an iterator the `_M_end` of the same sequence in their second entry.

Ties are broken arbitrarily. See `stable_multiway_merge` for a variant that breaks ties by sequence number but is slower.

The first entries of the pairs (i.e. the begin iterators) will be moved forward accordingly.

The output sequence has to provide enough space for all elements that are written to it.

This function will merge the input sequences:

- not stable
- parallel, depending on the input size and Settings
- using sampling for splitting
- using sentinels

You have to take care that the element the `_M_end` iterator points to is readable and contains a value that is greater than any other non-sentinel value in all sequences.

Example:

```
int sequences[10][11];
for (int __i = 0; __i < 10; ++__i)
    for (int __j = 0; __j < 11; ++__j)
        sequences[__i][__j] = __j; // __last one is sentinel!

int __out[33];
std::vector<std::pair<int*> > seqs;
for (int __i = 0; __i < 10; ++__i)
    { seqs.push(std::make_pair<int*>(sequences[__i],
                                    sequences[__i] + 10)) }

multiway_merge(seqs.begin(), seqs.end(), __target, std::less<int>(), 33);
```

Precondition

All input sequences must be sorted.

Target must provide enough space to merge out length elements or the number of elements in all sequences, whichever is smaller.

For each `__i`, `__seqs_begin[__i].second` must be the end marker of the sequence, but also reference the one more `__sentinel` element.

Postcondition

`[__target, return __value)` contains merged `__elements` from the input sequences.

`return __value - __target = min(__length, number of elements in all sequences).`

See also

`stable_multiway_merge_sentinels`

Template Parameters

<code>_RAIterIterator</code>	iterator over sequence of pairs of iterators
<code>_RAIterOut</code>	iterator over target sequence
<code>_DifferenceTp</code>	difference type for the sequence
<code>_Compare</code>	strict weak ordering type to compare elements in sequences

Parameters

<code>__seqs_begin</code>	<code>__begin</code> of sequence <code>__sequence</code>
<code>__seqs_end</code>	<code>_M_end</code> of sequence <code>__sequence</code>
<code>__target</code>	target sequence to merge to.
<code>__comp</code>	strict weak ordering to use for element comparison.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

`_M_end` iterator of output sequence

References [__sequential_multiway_merge\(\)](#), and [_GLIBCXX_CALL](#).

parallel_multiway_merge()

```
template<bool __stable, bool __sentinels, typename _RAIterIterator , typename _RAIter3 , typename
_DifferenceTp , typename _Splitter , typename _Compare >
_RAIter3 __gnu_parallel::parallel_multiway_merge (
    _RAIterIterator __seqs_begin,
    _RAIterIterator __seqs_end,
    _RAIter3 __target,
    _Splitter __splitter,
    _DifferenceTp __length,
    _Compare __comp,
    _ThreadIndex __num_threads )
```

Parallel multi-way merge routine.

The `_GLIBCXX_PARALLEL_DECISION` is based on the branching factor and runtime settings. Must not be called if the number of sequences is 1.

Template Parameters

<code>_Splitter</code>	functor to split input (either <code>__exact</code> or sampling based)
<code>__stable</code>	Stable merging incurs a performance penalty.
<code>__sentinel</code>	Ignored.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

End iterator of output sequence.

References [__is_sorted\(\)](#), [_GLIBCXX_CALL](#), [_GLIBCXX_PARALLEL_LENGTH](#), [__gnu_parallel::_Settings::get\(\)](#), and [__gnu_parallel::_Settings::merge_oversampling](#).

Referenced by [__parallel_merge_advance\(\)](#).

`parallel_sort_mwms()`

```
template<bool __stable, bool __exact, typename _RAIter, typename _Compare>
void __gnu_parallel::parallel_sort_mwms (
    _RAIter __begin,
    _RAIter __end,
    _Compare __comp,
    _ThreadIndex __num_threads )
```

PMWMS main call.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads to use.

References [_GLIBCXX_CALL](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_num_threads](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_pieces](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_samples](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_source](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_starts](#), [__gnu_parallel::PMWMSortingData<_RAIter>::M_temporary](#), [__gnu_parallel::_Settings::get\(\)](#), and [__gnu_parallel::_Settings::sort](#).

`parallel_sort_mwms_pu()`

```
template<bool __stable, bool __exact, typename _RAIter, typename _Compare>
void __gnu_parallel::parallel_sort_mwms_pu (
```



```

    _PMWMSortingData< _RAIter > * __sd,
    _Compare & __comp )

```

PMWMS code executed by each thread.

Parameters

<code>__sd</code>	Pointer to algorithm data.
<code>__comp</code>	Comparator.

References [__gnu_parallel::_PMWMSortingData< _RAIter >::M_num_threads](#), [__gnu_parallel::_PMWMSortingData< _RAIter >::__gnu_parallel::_PMWMSortingData< _RAIter >::M_source](#), [__gnu_parallel::_PMWMSortingData< _RAIter >::M_starts](#), [__gnu_parallel::_PMWMSortingData< _RAIter >::M_temporary](#), [__gnu_parallel::_Settings::get\(\)](#), [__gnu_parallel::_Settings::sort_mw](#) and [std::uninitialized_copy\(\)](#).

4.6.5 Variable Documentation

`_CASable_bits`

```
const int __gnu_parallel::_CASable_bits [static]
```

Number of bits of `_CASable`.

Referenced by [__decode2\(\)](#), and [__encode2\(\)](#).

`_CASable_mask`

```
const _CASable __gnu_parallel::_CASable_mask [static]
```

`_CASable` with the right half of bits set to 1.

Referenced by [__decode2\(\)](#).

4.7 `__gnu_pbds` Namespace Reference

Classes

- struct [associative_tag](#)
- class [basic_branch](#)
- struct [basic_branch_tag](#)
- class [basic_hash_table](#)
- struct [basic_hash_tag](#)
- struct [basic_invalidation_guarantee](#)
- struct [binary_heap_tag](#)
- struct [binomial_heap_tag](#)
- class [cc_hash_max_collision_check_resize_trigger](#)
- class [cc_hash_table](#)
- struct [cc_hash_tag](#)
- struct [container_error](#)
- struct [container_tag](#)
- struct [container_traits](#)
- struct [container_traits_base](#)
- struct [container_traits_base< binary_heap_tag >](#)
- struct [container_traits_base< binomial_heap_tag >](#)
- struct [container_traits_base< cc_hash_tag >](#)
- struct [container_traits_base< gp_hash_tag >](#)
- struct [container_traits_base< list_update_tag >](#)
- struct [container_traits_base< ov_tree_tag >](#)

- struct `container_traits_base< pairing_heap_tag >`
- struct `container_traits_base< pat_trie_tag >`
- struct `container_traits_base< rb_tree_tag >`
- struct `container_traits_base< rc_binomial_heap_tag >`
- struct `container_traits_base< splay_tree_tag >`
- struct `container_traits_base< thin_heap_tag >`
- class `direct_mask_range_hashing`
- class `direct_mod_range_hashing`
- class `gp_hash_table`
- struct `gp_hash_tag`
- class `hash_exponential_size_policy`
- class `hash_load_check_resize_trigger`
- class `hash_prime_size_policy`
- class `hash_standard_resize_policy`
- struct `insert_error`
- struct `join_error`
- class `linear_probe_fn`
- class `list_update`
- struct `list_update_tag`
- class `lu_counter_policy`
- class `lu_move_to_front_policy`
- struct `null_node_update`
- struct `null_type`
- struct `ov_tree_tag`
- struct `pairing_heap_tag`
- struct `pat_trie_tag`
- struct `point_invalidation_guarantee`
- class `priority_queue`
- struct `priority_queue_tag`
- class `quadratic_probe_fn`
- struct `range_invalidation_guarantee`
- struct `rb_tree_tag`
- struct `rc_binomial_heap_tag`
- struct `resize_error`
- class `sample_probe_fn`
- class `sample_range_hashing`
- class `sample_ranged_hash_fn`
- class `sample_ranged_probe_fn`
- class `sample_resize_policy`
- class `sample_resize_trigger`
- class `sample_size_policy`
- class `sample_tree_node_update`
- struct `sample_trie_access_traits`
- class `sample_trie_node_update`
- struct `sample_update_policy`
- struct `sequence_tag`
- struct `splay_tree_tag`
- struct `string_tag`
- struct `thin_heap_tag`
- class `tree`
- class `tree_order_statistics_node_update`

- struct [tree_tag](#)
- class [trie](#)
- class [trie_order_statistics_node_update](#)
- class [trie_prefix_search_node_update](#)
- struct [trie_string_access_traits](#)
- struct [trie_tag](#)
- struct [trivial_iterator_tag](#)

Typedefs

- typedef void [trivial_iterator_difference_type](#)

Functions

- void [__throw_container_error](#) ()
- void [__throw_insert_error](#) ()
- void [__throw_join_error](#) ()
- void [__throw_resize_error](#) ()

4.7.1 Detailed Description

GNU extensions for policy-based data structures for public use.

4.8 [__gnu_sequential](#) Namespace Reference

4.8.1 Detailed Description

GNU sequential classes for public use.

4.9 [abi](#) Namespace Reference

4.9.1 Detailed Description

The cross-vendor C++ Application Binary Interface. A namespace alias to [__cxxabiv1](#), but user programs should use the alias 'abi'.

A brief overview of an ABI is given in the libstdc++ FAQ, question 5.8 (you may have a copy of the FAQ locally, or you can view the online version at http://gcc.gnu.org/onlinedocs/libstdc++/faq.html#5_8).

GCC subscribes to a cross-vendor ABI for C++, sometimes called the IA64 ABI because it happens to be the native ABI for that platform. It is summarized at <http://www.codesourcery.com/cxx-abi/> along with the current specification.

For users of GCC greater than or equal to 3.x, entry points are available in `<cxxabi.h>`, which notes, *'It is not normally necessary for user programs to include this header, or use the entry points directly. However, this header is available should that be needed.'*

4.10 [std](#) Namespace Reference

Namespaces

- namespace [__debug](#)
- namespace [__detail](#)
- namespace [__parallel](#)
- namespace [chrono](#)
- namespace [decimal](#)
- namespace [experimental](#)
- namespace [filesystem](#)

- namespace [literals](#)
- namespace [numbers](#)
- namespace [placeholders](#)
- namespace [regex_constants](#)
- namespace [rel_ops](#)
- namespace [this_thread](#)
- namespace [tr1](#)
- namespace [tr2](#)

Classes

- struct [__atomic_base](#)
- struct [__atomic_base< _PTp * >](#)
- struct [__atomic_flag_base](#)
- class [__basic_future](#)
- class [__codecvt_abstract_base](#)
- class [__ctype_abstract_base](#)
- struct [__is_location_invariant](#)
- struct [__is_nullptr_t](#)
- class [__new_allocator](#)
- struct [__numeric_limits_base](#)
- struct [_Base_bitset](#)
- struct [_Base_bitset< 0 >](#)
- struct [_Base_bitset< 1 >](#)
- class [_Bind](#)
- class [_Bind_result](#)
- class [_Deque_base](#)
- struct [_Deque_iterator](#)
- class [_Function_base](#)
- struct [_Fwd_list_base](#)
- struct [_Fwd_list_const_iterator](#)
- struct [_Fwd_list_iterator](#)
- struct [_Fwd_list_node](#)
- struct [_Fwd_list_node_base](#)
- class [_List_base](#)
- struct [_List_const_iterator](#)
- struct [_List_iterator](#)
- struct [_List_node](#)
- class [_Node_handle](#)
- class [_Node_handle< _Value, _Value, _NodeAlloc >](#)
- class [_Node_handle_common](#)
- struct [_Node_insert_return](#)
- class [_Not_fn](#)
- struct [_Optional_base](#)
- struct [_Placeholder](#)
- struct [_Sp_ebo_helper< _Nm, _Tp, false >](#)
- struct [_Sp_ebo_helper< _Nm, _Tp, true >](#)
- class [_Temporary_buffer](#)
- struct [_Tuple_impl](#)
- struct [_Tuple_impl< _Idx, _Head, _Tail... >](#)
- struct [_Vector_base](#)

- struct [add_const](#)
- struct [add_cv](#)
- struct [add_lvalue_reference](#)
- struct [add_pointer](#)
- struct [add_rvalue_reference](#)
- struct [add_volatile](#)
- struct [adopt_lock_t](#)
- struct [aligned_storage](#)
- struct [aligned_union](#)
- struct [alignment_of](#)
- class [allocator](#)
- class [allocator< void >](#)
- struct [allocator_traits](#)
- struct [allocator_traits< allocator< _Tp > >](#)
- struct [allocator_traits< allocator< void > >](#)
- struct [allocator_traits< pmr::polymorphic_allocator< _Tp > >](#)
- class [any](#)
- struct [array](#)
- class [atomic](#)
- struct [atomic< _Tp * >](#)
- struct [atomic< bool >](#)
- struct [atomic< char >](#)
- struct [atomic< char16_t >](#)
- struct [atomic< char32_t >](#)
- struct [atomic< int >](#)
- struct [atomic< long >](#)
- struct [atomic< long long >](#)
- struct [atomic< short >](#)
- struct [atomic< signed char >](#)
- struct [atomic< unsigned char >](#)
- struct [atomic< unsigned int >](#)
- struct [atomic< unsigned long >](#)
- struct [atomic< unsigned long long >](#)
- struct [atomic< unsigned short >](#)
- struct [atomic< wchar_t >](#)
- struct [atomic_flag](#)
- struct [atomic_ref](#)
- class [auto_ptr](#)
- struct [auto_ptr_ref](#)
- class [back_insert_iterator](#)
- class [bad_alloc](#)
- class [bad_any_cast](#)
- class [bad_cast](#)
- class [bad_exception](#)
- class [bad_function_call](#)
- class [bad_optional_access](#)
- class [bad_typeid](#)
- class [bad_weak_ptr](#)
- class [basic_filebuf](#)
- class [basic_fstream](#)
- class [basic_ifstream](#)

- class `basic_ios`
- class `basic_iostream`
- class `basic_istream`
- class `basic_istreamstream`
- class `basic_ofstream`
- class `basic_ostream`
- class `basic_ostreamstream`
- class `basic_regex`
- class `basic_streambuf`
- class `basic_string`
- class `basic_string_view`
- class `basic_stringbuf`
- class `basic_stringstream`
- class `bernoulli_distribution`
- struct `bidirectional_iterator_tag`
- struct `binary_function`
- class `binary_negate`
- class `binder1st`
- class `binder2nd`
- class `binomial_distribution`
- class `bitset`
- class `cauchy_distribution`
- struct `char_traits`
- struct `char_traits< __gnu_cxx::character< _Value, _Int, _St > >`
- struct `char_traits< char >`
- struct `char_traits< wchar_t >`
- class `chi_squared_distribution`
- class `codecvt`
- class `codecvt< _InternT, _ExternT, encoding_state >`
- class `codecvt< char, char, mbstate_t >`
- class `codecvt< char16_t, char, mbstate_t >`
- class `codecvt< char32_t, char, mbstate_t >`
- class `codecvt< wchar_t, char, mbstate_t >`
- class `codecvt_base`
- class `codecvt_byname`
- class `collate`
- class `collate_byname`
- class `common_iterator`
- struct `common_type`
- struct `common_type< chrono::duration< _Rep, _Period > >`
- struct `common_type< chrono::duration< _Rep, _Period >, chrono::duration< _Rep, _Period > >`
- struct `common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > >`
- struct `common_type< chrono::time_point< _Clock, _Duration > >`
- struct `common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >`
- struct `common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >`
- struct `compare_three_way_result`
- class `complex`
- class `complex< double >`
- class `complex< float >`
- class `complex< long double >`
- class `condition_variable`

- class [condition_variable_any](#)
- struct [conditional](#)
- class [const_mem_fun1_ref_t](#)
- class [const_mem_fun1_t](#)
- class [const_mem_fun_ref_t](#)
- class [const_mem_fun_t](#)
- struct [contiguous_iterator_tag](#)
- class [counted_iterator](#)
- class [ctype](#)
- class [ctype< char >](#)
- class [ctype< wchar_t >](#)
- struct [ctype_base](#)
- class [ctype_byname](#)
- class [ctype_byname< char >](#)
- class [decay](#)
- struct [default_delete](#)
- struct [default_delete< _Tp\[\]>](#)
- struct [default_sentinel_t](#)
- struct [defer_lock_t](#)
- class [deque](#)
- struct [destroying_delete_t](#)
- class [discard_block_engine](#)
- class [discrete_distribution](#)
- struct [divides](#)
- struct [divides< void >](#)
- class [domain_error](#)
- struct [enable_if](#)
- class [enable_shared_from_this](#)
- struct [equal_to](#)
- struct [equal_to< void >](#)
- class [error_category](#)
- class [error_code](#)
- class [error_condition](#)
- class [exception](#)
- class [exponential_distribution](#)
- struct [extent](#)
- class [extreme_value_distribution](#)
- class [fisher_f_distribution](#)
- struct [forward_iterator_tag](#)
- class [forward_list](#)
- class [fpos](#)
- struct [from_chars_result](#)
- class [front_insert_iterator](#)
- class [function< _Res\(_ArgTypes...\)>](#)
- class [future](#)
- class [future< _Res & >](#)
- class [future< void >](#)
- class [future_error](#)
- class [gamma_distribution](#)
- class [geometric_distribution](#)
- struct [greater](#)

- struct `greater< void >`
- struct `greater_equal`
- struct `greater_equal< void >`
- class `gslice`
- class `gslice_array`
- struct `has_unique_object_representations`
- struct `has_virtual_destructor`
- struct `hash`
- struct `hash< __debug::bitset< _Nb > >`
- struct `hash< __debug::vector< bool, _Alloc > >`
- struct `hash< __gnu_cxx::__u16vstring >`
- struct `hash< __gnu_cxx::__u32vstring >`
- struct `hash< __gnu_cxx::__vstring >`
- struct `hash< __gnu_cxx::__wvstring >`
- struct `hash< __gnu_cxx::throw_value_limit >`
- struct `hash< __gnu_cxx::throw_value_random >`
- struct `hash< __gnu_debug::basic_string< _CharT > >`
- struct `hash< __shared_ptr< _Tp, _Lp > >`
- struct `hash< _Tp * >`
- struct `hash< bool >`
- struct `hash< char >`
- struct `hash< char16_t >`
- struct `hash< char32_t >`
- struct `hash< double >`
- struct `hash< error_code >`
- struct `hash< error_condition >`
- struct `hash< experimental::optional< _Tp > >`
- struct `hash< experimental::shared_ptr< _Tp > >`
- struct `hash< float >`
- struct `hash< int >`
- struct `hash< long >`
- struct `hash< long double >`
- struct `hash< long long >`
- struct `hash< shared_ptr< _Tp > >`
- struct `hash< short >`
- struct `hash< signed char >`
- struct `hash< string >`
- struct `hash< thread::id >`
- struct `hash< type_index >`
- struct `hash< u16string >`
- struct `hash< u32string >`
- struct `hash< unique_ptr< _Tp, _Dp > >`
- struct `hash< unsigned char >`
- struct `hash< unsigned int >`
- struct `hash< unsigned long >`
- struct `hash< unsigned long long >`
- struct `hash< unsigned short >`
- struct `hash< wchar_t >`
- struct `hash< wstring >`
- struct `hash<::bitset< _Nb > >`
- struct `hash<::vector< bool, _Alloc > >`

- struct [identity](#)
- class [independent_bits_engine](#)
- class [indirect_array](#)
- class [initializer_list](#)
- struct [input_iterator_tag](#)
- class [insert_iterator](#)
- struct [integer_sequence](#)
- struct [integral_constant](#)
- class [invalid_argument](#)
- struct [invoke_result](#)
- class [ios_base](#)
- struct [is_abstract](#)
- struct [is_aggregate](#)
- struct [is_arithmetic](#)
- struct [is_array](#)
- struct [is_assignable](#)
- struct [is_base_of](#)
- struct [is_bind_expression](#)
- struct [is_bind_expression< _Bind< _Signature > >](#)
- struct [is_bind_expression< _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< const _Bind< _Signature > >](#)
- struct [is_bind_expression< const _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< const volatile _Bind< _Signature > >](#)
- struct [is_bind_expression< const volatile _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< volatile _Bind< _Signature > >](#)
- struct [is_bind_expression< volatile _Bind_result< _Result, _Signature > >](#)
- struct [is_bounded_array](#)
- struct [is_class](#)
- struct [is_compound](#)
- struct [is_const](#)
- struct [is_constructible](#)
- struct [is_convertible](#)
- struct [is_copy_assignable](#)
- struct [is_copy_constructible](#)
- struct [is_default_constructible](#)
- struct [is_destructible](#)
- struct [is_empty](#)
- struct [is_enum](#)
- struct [is_error_code_enum](#)
- struct [is_error_code_enum< future_errc >](#)
- struct [is_error_condition_enum](#)
- struct [is_final](#)
- struct [is_floating_point](#)
- struct [is_function](#)
- struct [is_fundamental](#)
- struct [is_integral](#)
- struct [is_invocable](#)
- struct [is_invocable_r](#)
- struct [is_layout_compatible](#)
- struct [is_literal_type](#)
- struct [is_lvalue_reference](#)

- struct [is_member_function_pointer](#)
- struct [is_member_object_pointer](#)
- struct [is_member_pointer](#)
- struct [is_move_assignable](#)
- struct [is_move_constructible](#)
- struct [is_nothrow_assignable](#)
- struct [is_nothrow_constructible](#)
- struct [is_nothrow_convertible](#)
- struct [is_nothrow_copy_assignable](#)
- struct [is_nothrow_copy_constructible](#)
- struct [is_nothrow_default_constructible](#)
- struct [is_nothrow_destructible](#)
- struct [is_nothrow_invocable](#)
- struct [is_nothrow_invocable_r](#)
- struct [is_nothrow_move_assignable](#)
- struct [is_nothrow_move_constructible](#)
- struct [is_nothrow_swappable](#)
- struct [is_nothrow_swappable_with](#)
- struct [is_null_pointer](#)
- struct [is_object](#)
- struct [is_placeholder](#)
- struct [is_placeholder< _Placeholder< _Num > >](#)
- struct [is_pod](#)
- struct [is_pointer](#)
- struct [is_pointer_interconvertible_base_of](#)
- struct [is_polymorphic](#)
- struct [is_reference](#)
- struct [is_rvalue_reference](#)
- struct [is_same](#)
- struct [is_scalar](#)
- struct [is_signed](#)
- struct [is_standard_layout](#)
- struct [is_swappable](#)
- struct [is_swappable_with](#)
- struct [is_trivial](#)
- struct [is_trivially_assignable](#)
- struct [is_trivially_constructible](#)
- struct [is_trivially_copy_assignable](#)
- struct [is_trivially_copy_constructible](#)
- struct [is_trivially_copyable](#)
- struct [is_trivially_default_constructible](#)
- struct [is_trivially_destructible](#)
- struct [is_trivially_move_assignable](#)
- struct [is_trivially_move_constructible](#)
- struct [is_unbounded_array](#)
- struct [is_union](#)
- struct [is_unsigned](#)
- struct [is_void](#)
- struct [is_volatile](#)
- class [istream_iterator](#)
- class [istreambuf_iterator](#)

- struct `iterator`
- struct `iterator_traits`
- struct `iterator_traits< _Tp * >`
- class `jthread`
- class `length_error`
- struct `less`
- struct `less< void >`
- struct `less_equal`
- struct `less_equal< void >`
- class `linear_congruential_engine`
- class `list`
- class `locale`
- class `lock_guard`
- class `logic_error`
- struct `logical_and`
- struct `logical_and< void >`
- struct `logical_not`
- struct `logical_not< void >`
- struct `logical_or`
- struct `logical_or< void >`
- class `lognormal_distribution`
- struct `make_signed`
- struct `make_unsigned`
- class `map`
- class `mask_array`
- class `match_results`
- class `mem_fun1_ref_t`
- class `mem_fun1_t`
- class `mem_fun_ref_t`
- class `mem_fun_t`
- class `mersenne_twister_engine`
- class `messages`
- struct `messages_base`
- class `messages_byname`
- struct `minus`
- struct `minus< void >`
- struct `modulus`
- struct `modulus< void >`
- class `money_base`
- class `money_get`
- class `money_put`
- class `moneypunct`
- class `moneypunct_byname`
- class `move_iterator`
- class `move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>`
- class `multimap`
- struct `multiplies`
- struct `multiplies< void >`
- class `multiset`
- class `mutex`
- struct `negate`

- struct `negate< void >`
- class `negative_binomial_distribution`
- class `nested_exception`
- class `normal_distribution`
- struct `nostopstate_t`
- struct `not_equal_to`
- struct `not_equal_to< void >`
- struct `nullopt_t`
- class `num_get`
- class `num_put`
- struct `numeric_limits`
- struct `numeric_limits< bool >`
- struct `numeric_limits< char >`
- struct `numeric_limits< char16_t >`
- struct `numeric_limits< char32_t >`
- struct `numeric_limits< double >`
- struct `numeric_limits< float >`
- struct `numeric_limits< int >`
- struct `numeric_limits< long >`
- struct `numeric_limits< long double >`
- struct `numeric_limits< long long >`
- struct `numeric_limits< short >`
- struct `numeric_limits< signed char >`
- struct `numeric_limits< unsigned char >`
- struct `numeric_limits< unsigned int >`
- struct `numeric_limits< unsigned long >`
- struct `numeric_limits< unsigned long long >`
- struct `numeric_limits< unsigned short >`
- struct `numeric_limits< wchar_t >`
- class `num_punct`
- class `num_punct_byname`
- struct `once_flag`
- class `optional`
- class `ostream_iterator`
- class `ostreambuf_iterator`
- class `out_of_range`
- struct `output_iterator_tag`
- class `overflow_error`
- struct `owner_less`
- struct `owner_less< shared_ptr< _Tp > >`
- struct `owner_less< void >`
- struct `owner_less< weak_ptr< _Tp > >`
- class `packaged_task< _Res(_ArgTypes...)>`
- struct `pair`
- class `piecewise_constant_distribution`
- struct `piecewise_construct_t`
- class `piecewise_linear_distribution`
- struct `plus`
- class `pointer_to_binary_function`
- class `pointer_to_unary_function`
- struct `pointer_traits`

- struct [pointer_traits< _Tp * >](#)
- class [poisson_distribution](#)
- class [priority_queue](#)
- struct [projected](#)
- class [promise](#)
- class [promise< _Res & >](#)
- class [promise< void >](#)
- class [queue](#)
- struct [random_access_iterator_tag](#)
- class [random_device](#)
- class [range_error](#)
- struct [rank](#)
- struct [ratio](#)
- struct [ratio_equal](#)
- struct [ratio_greater](#)
- struct [ratio_greater_equal](#)
- struct [ratio_less](#)
- struct [ratio_less_equal](#)
- struct [ratio_not_equal](#)
- class [raw_storage_iterator](#)
- class [recursive_mutex](#)
- class [recursive_timed_mutex](#)
- class [reference_wrapper](#)
- class [regex_error](#)
- class [regex_iterator](#)
- class [regex_token_iterator](#)
- class [regex_traits](#)
- struct [remove_all_extents](#)
- struct [remove_const](#)
- struct [remove_cv](#)
- struct [remove_extent](#)
- struct [remove_pointer](#)
- struct [remove_reference](#)
- struct [remove_volatile](#)
- struct [result_of](#)
- class [reverse_iterator](#)
- class [runtime_error](#)
- class [scoped_allocator_adaptor](#)
- class [scoped_lock](#)
- class [seed_seq](#)
- class [set](#)
- class [shared_future](#)
- class [shared_future< _Res & >](#)
- class [shared_future< void >](#)
- class [shared_lock](#)
- class [shared_mutex](#)
- class [shared_ptr](#)
- class [shared_timed_mutex](#)
- class [shuffle_order_engine](#)
- class [slice](#)
- class [slice_array](#)

- struct `source_location`
- class `stack`
- class `stop_callback`
- class `stop_source`
- class `stop_token`
- class `student_t_distribution`
- class `sub_match`
- class `subtract_with_carry_engine`
- class `system_error`
- class `thread`
- class `time_base`
- class `time_get`
- class `time_get_byname`
- class `time_put`
- class `time_put_byname`
- class `timed_mutex`
- struct `to_chars_result`
- struct `try_to_lock_t`
- class `tuple`
- class `tuple<_T1, _T2>`
- struct `tuple_element`
- struct `tuple_element<0, pair<_Tp1, _Tp2>>`
- struct `tuple_element<1, pair<_Tp1, _Tp2>>`
- struct `tuple_element<__i, tuple<_Types...>>`
- struct `tuple_element<_Ind, array<_Tp, _Nm>>`
- struct `tuple_size`
- struct `tuple_size<array<_Tp, _Nm>>`
- struct `tuple_size<pair<_Tp1, _Tp2>>`
- struct `tuple_size<tuple<_Elements...>>`
- struct `type_index`
- class `type_info`
- struct `unary_function`
- class `unary_negate`
- class `underflow_error`
- struct `underlying_type`
- class `uniform_int_distribution`
- class `uniform_real_distribution`
- class `unique_lock`
- class `unique_ptr`
- class `unique_ptr<_Tp[], _Dp>`
- class `unordered_map`
- class `unordered_multimap`
- class `unordered_multiset`
- class `unordered_set`
- struct `unwrap_ref_decay`
- struct `unwrap_reference`
- struct `uses_allocator`
- struct `uses_allocator<tuple<_Types...>, _Alloc>`
- class `valarray`
- class `vector`
- class `vector<bool, _Alloc>`

- class [wbuffer_convert](#)
- class [weak_ptr](#)
- class [weibull_distribution](#)
- class [wstring_convert](#)

Concepts

- concept [same_as](#)
- concept [derived_from](#)
- concept [convertible_to](#)
- concept [common_reference_with](#)
- concept [common_with](#)
- concept [assignable_from](#)
- concept [destructible](#)
- concept [constructible_from](#)
- concept [default_initializable](#)
- concept [move_constructible](#)
- concept [copy_constructible](#)
- concept [invocable](#)
- concept [regular_invocable](#)
- concept [predicate](#)
- concept [relation](#)
- concept [equivalence_relation](#)
- concept [strict_weak_order](#)
- concept [indirectly_readable](#)
- concept [indirectly_writable](#)
- concept [weakly_incrementable](#)
- concept [indirectly_movable](#)
- concept [indirectly_copyable](#)
- concept [indirectly_swappable](#)
- concept [indirectly_comparable](#)
- concept [permutable](#)
- concept [mergeable](#)
- concept [sortable](#)
- concept [uniform_random_bit_generator](#)

Typedefs

- `template<typename _Tp >`
 using [__allocator_base](#) = [__new_allocator](#)< _Tp >
- `template<typename _Tp, typename _Up >`
 using [__assigns_from_optional](#) = `__or_< is_assignable< _Tp &, const optional< _Up > & >, is_assignable< _Tp &, optional< _Up > & >, is_assignable< _Tp &, const optional< _Up > && >, is_assignable< _Tp &, optional< _Up > && > >`
- `typedef unsigned char __atomic_flag_data_type`
- `template<typename _RAIter, typename _Hash, typename _Pred, typename _Val = typename iterator_traits<_RAIter>::value_type, typename _Diff = typename iterator_traits<_RAIter>::difference_type>`
 using [__boyer_moore_base_t](#) = `__conditional_t< is_byte_like< _Val, _Pred >::value, boyer_moore_map_base< _Diff, 256, _Pred >, boyer_moore_map_base< _Val, _Diff, _Hash, _Pred > >`
- `template<typename _IntegerType >`
 using [__byte_op_t](#) = `typename __byte_operand< _IntegerType >::__type`
- `typedef FILE __c_file`

- typedef `__locale_t` `__c_locale`
- typedef `__pthread_mutex_t` `__c_lock`
- template<bool `_Cond`, typename `_If`, typename `_Else` >
using `__conditional_t` = typename `__conditional<_Cond>::template type<_If, _Else>`
- template<typename `_Tp`, typename `_Up` >
using `__converts_from_optional` = `__or< is_constructible<_Tp, const optional<_Up> & >, is_constructible<_Tp, optional<_Up> & >, is_constructible<_Tp, const optional<_Up> && >, is_constructible<_Tp, optional<_Up> && >, is_convertible<const optional<_Up> &, _Tp>, is_convertible<optional<_Up> &, _Tp>, is_convertible<const optional<_Up> &&, _Tp>, is_convertible<optional<_Up> &&, _Tp>>`
- typedef `basic_string<char>` `__cow_string`
- template<typename `_Tp` >
using `__empty_not_final` = `__conditional_t<__is_final(_Tp), false_type, __is_empty_non_tuple<_Tp>>`
- template<typename `_Tp`, typename `_Up` = typename `remove_cv<_Tp>::type`, typename `enable_if<is_same<_Tp, _Up><::value>::type, size_t = tuple_size<_Tp>::value>` >
using `__enable_if_has_tuple_size` = `_Tp`
- template<typename `_Func`, typename `_SfinaeType` >
using `__has_is_transparent_t` = typename `__has_is_transparent<_Func, _SfinaeType>::type`
- template<typename `_ToElementType`, typename `_FromElementType` >
using `__is_array_convertible` = `is_convertible<_FromElementType(*)[], _ToElementType(*)[]>`
- template<typename `_Tp` >
using `__is_in_place_type` = `bool_constant<__is_in_place_type_v<_Tp>>`
- template<typename `_Tp`, typename `_Up` >
using `__is_nothrow_assignable_impl` = `__bool_constant<__is_nothrow_assignable(_Tp, _Up)>`
- template<typename `_Tp`, typename `_Tp2` = typename `decay<_Tp>::type` >
using `__is_socketlike` = `__or< is_integral<_Tp2>, is_enum<_Tp2>>`
- template<typename `_InputIterator` >
using `__iter_key_t` = `remove_const_t<typename iterator_traits<_InputIterator>::value_type::first_type>`
- template<typename `_InputIterator` >
using `__iter_to_alloc_t` = `pair<add_const_t<__iter_key_t<_InputIterator>>, __iter_val_t<_InputIterator>>`
- template<typename `_InputIterator` >
using `__iter_val_t` = typename `iterator_traits<_InputIterator>::value_type::second_type`
- template<typename `_Iter` >
using `__iterator_category_t` = typename `iterator_traits<_Iter>::iterator_category`
- template<typename `_Tp`, typename `_Up` >
using `__optional_eq_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())==std::declval<const _Up &>())>`
- template<typename `_Tp`, typename `_Up` >
using `__optional_ge_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())>=std::declval<const _Up &>())>`
- template<typename `_Tp`, typename `_Up` >
using `__optional_gt_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())>std::declval<const _Up &>())>`
- template<typename `_Tp`, typename `_Up` >
using `__optional_le_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())<=std::declval<const _Up &>())>`
- template<typename `_Tp`, typename `_Up` >
using `__optional_lt_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())<std::declval<const _Up &>())>`
- template<typename `_Tp`, typename `_Up` >
using `__optional_ne_t` = `__optional_relop_t<decltype(std::declval<const _Tp &>())!=std::declval<const _Up &>())>`

- `template<typename _Tp >`
`using __optional_relop_t = enable_if_t< is_convertible< _Tp, bool >::value, bool >`
- `template<typename _Ptr, typename _Tp >`
`using __ptr_rebind = typename pointer_traits< _Ptr >::template rebind< _Tp >`
- `template<typename _Ptr >`
`using __ptr_traits_elem_t = typename __get_first_arg< _Ptr >::type`
- `template<typename _Is, typename _Tp >`
`using __rvalue_stream_extraction_t = _Is &&`
- `template<typename _Os, typename _Tp >`
`using __rvalue_stream_insertion_t = _Os &&`
- `using __semaphore_impl = __atomic_semaphore`
- `typedef basic_string< char > __sso_string`
- `template<size_t __i, typename _Tp >`
`using __tuple_element_t = typename tuple_element< __i, _Tp >::type`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc =`
`std::allocator<std::pair<const _Key, _Tp> >, typename _Tr = __umap_traits<__cache_default<_Key, _Hash>::value>>`
`using __umap_hashtable = _Hashtable< _Key, std::pair< const _Key, _Tp >, _Alloc, __detail::_Select1st, ↵`
`_Pred, _Hash, __detail::_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy,`
`_Tr >`
- `template<bool _Cache>`
`using __umap_traits = __detail::_Hashtable_traits< _Cache, false, true >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc =`
`std::allocator<std::pair<const _Key, _Tp> >, typename _Tr = __ummap_traits<__cache_default<_Key, _Hash>::value>>`
`using __ummap_hashtable = _Hashtable< _Key, std::pair< const _Key, _Tp >, _Alloc, __detail::_Select1st, ↵`
`_Pred, _Hash, __detail::_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy,`
`_Tr >`
- `template<bool _Cache>`
`using __ummap_traits = __detail::_Hashtable_traits< _Cache, false, false >`
- `template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::↵`
`::allocator<_Value>, typename _Tr = __umset_traits<__cache_default<_Value, _Hash>::value>>`
`using __umset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail::_Identity, _Pred, _Hash, __detail::_↵`
`_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy, _Tr >`
- `template<bool _Cache>`
`using __umset_traits = __detail::_Hashtable_traits< _Cache, true, false >`
- `template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::↵`
`::allocator<_Value>, typename _Tr = __uset_traits<__cache_default<_Value, _Hash>::value>>`
`using __uset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail::_Identity, _Pred, _Hash, __detail::_↵`
`_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy, _Tr >`
- `template<bool _Cache>`
`using __uset_traits = __detail::_Hashtable_traits< _Cache, true, true >`
- `template<typename _Fn, typename... _Args>`
`using __Bind_front_t = __Bind_front< decay_t< _Fn >, decay_t< _Args >... >`
- `typedef unsigned long __Bit_type`
- `template<typename _Path, typename _Result = _Path, typename _Path2 = decltype(std::declval<_Path>().make_preferred().↵`
`filename())>`
`using __If_fs_path = enable_if_t< is_same_v< _Path, _Path2 >, _Result >`
- `template<typename _Alloc >`
`using __RequireAllocator = typename enable_if< __is_allocator< _Alloc >::value, _Alloc >::type`
- `template<typename _InIter >`
`using __RequireInputIter = __enable_if_t< is_convertible< __iterator_category_t< _InIter >, input_iterator_tag`
`>::value >`
- `template<typename _Alloc >`
`using __RequireNotAllocator = typename enable_if<!__is_allocator< _Alloc >::value, _Alloc >::type`

- `template<typename _Tp >`
 using `add_const_t` = typename `add_const`< `_Tp` >::type
- `template<typename _Tp >`
 using `add_cv_t` = typename `add_cv`< `_Tp` >::type
- `template<typename _Tp >`
 using `add_lvalue_reference_t` = typename `add_lvalue_reference`< `_Tp` >::type
- `template<typename _Tp >`
 using `add_pointer_t` = typename `add_pointer`< `_Tp` >::type
- `template<typename _Tp >`
 using `add_rvalue_reference_t` = typename `add_rvalue_reference`< `_Tp` >::type
- `template<typename _Tp >`
 using `add_volatile_t` = typename `add_volatile`< `_Tp` >::type
- `template<size_t _Len, size_t _Align = __alignof__(typename __aligned_storage_msa<_Len>::__type)>`
 using `aligned_storage_t` = typename `aligned_storage`< `_Len`, `_Align` >::type
- `template<size_t _Len, typename... _Types>`
 using `aligned_union_t` = typename `aligned_union`< `_Len`, `_Types...` >::type
- `typedef atomic< bool > atomic_bool`
- `typedef atomic< char > atomic_char`
- `typedef atomic< char16_t > atomic_char16_t`
- `typedef atomic< char32_t > atomic_char32_t`
- `typedef atomic< int > atomic_int`
- `typedef atomic< int16_t > atomic_int16_t`
- `typedef atomic< int32_t > atomic_int32_t`
- `typedef atomic< int64_t > atomic_int64_t`
- `typedef atomic< int8_t > atomic_int8_t`
- `typedef atomic< int_fast16_t > atomic_int_fast16_t`
- `typedef atomic< int_fast32_t > atomic_int_fast32_t`
- `typedef atomic< int_fast64_t > atomic_int_fast64_t`
- `typedef atomic< int_fast8_t > atomic_int_fast8_t`
- `typedef atomic< int_least16_t > atomic_int_least16_t`
- `typedef atomic< int_least32_t > atomic_int_least32_t`
- `typedef atomic< int_least64_t > atomic_int_least64_t`
- `typedef atomic< int_least8_t > atomic_int_least8_t`
- `typedef atomic< intmax_t > atomic_intmax_t`
- `typedef atomic< intptr_t > atomic_intptr_t`
- `typedef atomic< long long > atomic_llong`
- `typedef atomic< long > atomic_long`
- `typedef atomic< ptrdiff_t > atomic_ptrdiff_t`
- `typedef atomic< signed char > atomic_schar`
- `typedef atomic< short > atomic_short`
- `typedef atomic< size_t > atomic_size_t`
- `typedef atomic< unsigned char > atomic_uchar`
- `typedef atomic< unsigned int > atomic_uint`
- `typedef atomic< uint16_t > atomic_uint16_t`
- `typedef atomic< uint32_t > atomic_uint32_t`
- `typedef atomic< uint64_t > atomic_uint64_t`
- `typedef atomic< uint8_t > atomic_uint8_t`
- `typedef atomic< uint_fast16_t > atomic_uint_fast16_t`
- `typedef atomic< uint_fast32_t > atomic_uint_fast32_t`
- `typedef atomic< uint_fast64_t > atomic_uint_fast64_t`
- `typedef atomic< uint_fast8_t > atomic_uint_fast8_t`

- typedef `atomic< uint_least16_t > atomic_uint_least16_t`
- typedef `atomic< uint_least32_t > atomic_uint_least32_t`
- typedef `atomic< uint_least64_t > atomic_uint_least64_t`
- typedef `atomic< uint_least8_t > atomic_uint_least8_t`
- typedef `atomic< uintmax_t > atomic_uintmax_t`
- typedef `atomic< uintptr_t > atomic_uintptr_t`
- typedef `atomic< unsigned long long > atomic_ullong`
- typedef `atomic< unsigned long > atomic_ulong`
- typedef `atomic< unsigned short > atomic_ushort`
- typedef `atomic< wchar_t > atomic_wchar_t`
- typedef `ratio< 1, 1000000000000000000 > atto`
- using `binary_semaphore = std::counting_semaphore< 1 >`
- template<bool __v>
using `bool_constant = integral_constant< bool, __v >`
- typedef `ratio< 1, 100 > centi`
- typedef `match_results< const char * > cmatch`
- template<typename... _Ts>
using `common_comparison_category_t = typename common_comparison_category< _Ts... >::type`
- template<typename... _Tp>
using `common_reference_t = typename common_reference< _Tp... >::type`
- template<typename... _Tp>
using `common_type_t = typename common_type< _Tp... >::type`
- template<typename _Tp, typename _Up = _Tp>
using `compare_three_way_result_t = typename __detail::__cmp3way_res_impl< _Tp, _Up >::type`
- template<bool _Cond, typename _Iftrue, typename _Iffalse >
using `conditional_t = typename conditional< _Cond, _Iftrue, _Iffalse >::type`
- typedef `regex_iterator< const char * > cregex_iterator`
- typedef `regex_token_iterator< const char * > cregex_token_iterator`
- typedef `sub_match< const char * > csub_match`
- typedef `ratio< 10, 1 > deca`
- template<typename _Tp >
using `decay_t = typename decay< _Tp >::type`
- typedef `ratio< 1, 10 > deci`
- typedef `minstd_rand0 default_random_engine`
- template<bool _Cond, typename _Tp = void>
using `enable_if_t = typename enable_if< _Cond, _Tp >::type`
- typedef `ratio< 1000000000000000000, 1 > exa`
- using `false_type = integral_constant< bool, false >`
- typedef `ratio< 1, 1000000000000000000 > femto`
- typedef `basic_filebuf< char > filebuf`
- typedef `basic_fstream< char > fstream`
- typedef `ratio< 1000000000, 1 > giga`
- typedef `ratio< 100, 1 > hecto`
- typedef `basic_ifstream< char > ifstream`
- template<size_t... _Idx>
using `index_sequence = integer_sequence< size_t, _Idx... >`
- template<typename... _Types>
using `index_sequence_for = make_index_sequence< sizeof...(_Types)>`
- template<typename _Fn, typename... _Is>
using `indirect_result_t = invoke_result_t< _Fn, iter_reference_t< _Is >... >`
- template<typename _Fn, typename... _Args>
using `invoke_result_t = typename invoke_result< _Fn, _Args... >::type`

- typedef [basic_ios](#)< char > [ios](#)
- typedef [basic_iostream](#)< char > [iostream](#)
- typedef [basic_istream](#)< char > [istream](#)
- typedef [basic_istream](#)< char > [istream](#)
- typedef [basic_istream](#)< char > [istream](#)
- template<indirectly_readable _Tp>
using [iter_common_reference_t](#) = common_reference_t< iter_reference_t< _Tp >, iter_value_t< _Tp > & >
- template<typename _Tp >
using [iter_difference_t](#) = __detail::__iter_diff_t< [remove_cvref_t](#)< _Tp > >
- template<__detail::__dereferenceable _Tp>
using [iter_reference_t](#) = decltype(*std::declval< _Tp & >())
- template<__detail::__dereferenceable _Tp>
using [iter_rvalue_reference_t](#) = ranges::__cust_imove::__IMove::__type< _Tp & >
- template<typename _Tp >
using [iter_value_t](#) = __detail::__iter_value_t< [remove_cvref_t](#)< _Tp > >
- typedef [ratio](#)< 1000, 1 > [kilo](#)
- typedef [shuffle_order_engine](#)< [minstd_rand0](#), 256 > [knuth_b](#)
- template<size_t _Num>
using [make_index_sequence](#) = [make_integer_sequence](#)< size_t, _Num >
- template<typename _Tp, _Tp _Num>
using [make_integer_sequence](#) = __make_integer_seq< [integer_sequence](#), _Tp, _Num >
- template<typename _Tp >
using [make_signed_t](#) = typename [make_signed](#)< _Tp >::type
- template<typename _Tp >
using [make_unsigned_t](#) = typename [make_unsigned](#)< _Tp >::type
- typedef [ratio](#)< 1000000, 1 > [mega](#)
- typedef [ratio](#)< 1, 1000000 > [micro](#)
- typedef [ratio](#)< 1, 1000 > [milli](#)
- typedef [linear_congruential_engine](#)< uint_fast32_t, 48271UL, 0UL, 2147483647UL > [minstd_rand](#)
- typedef [linear_congruential_engine](#)< uint_fast32_t, 16807UL, 0UL, 2147483647UL > [minstd_rand0](#)
- typedef [mersenne_twister_engine](#)< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL > [mt19937](#)
- typedef [mersenne_twister_engine](#)< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL > [mt19937_64](#)
- typedef [ratio](#)< 1, 1000000000 > [nano](#)
- typedef void(* [new_handler](#)) ()
- typedef decltype(nullptr) [nullptr_t](#)
- typedef [basic_ofstream](#)< char > [ofstream](#)
- typedef [basic_ostream](#)< char > [ostream](#)
- typedef [basic_ostream](#)< char > [ostream](#)
- typedef [ratio](#)< 1000000000000000, 1 > [peta](#)
- typedef [ratio](#)< 1, 1000000000000 > [pico](#)
- typedef __PTRDIFF_TYPE__ [ptrdiff_t](#)
- typedef [discard_block_engine](#)< [ranlux24_base](#), 223, 23 > [ranlux24](#)
- typedef [subtract_with_carry_engine](#)< uint_fast32_t, 24, 10, 24 > [ranlux24_base](#)
- typedef [discard_block_engine](#)< [ranlux48_base](#), 389, 11 > [ranlux48](#)
- typedef [subtract_with_carry_engine](#)< uint_fast64_t, 48, 5, 12 > [ranlux48_base](#)
- template<typename _R1, typename _R2 >
using [ratio_add](#) = typename __ratio_add< _R1, _R2 >::type
- template<typename _R1, typename _R2 >
using [ratio_divide](#) = typename __ratio_divide< _R1, _R2 >::type

- `template<typename _R1, typename _R2 >`
`using ratio_multiply = typename __ratio_multiply< _R1, _R2 >::type`
- `template<typename _R1, typename _R2 >`
`using ratio_subtract = typename __ratio_subtract< _R1, _R2 >::type`
- `typedef basic_regex< char > regex`
- `template<typename _Tp >`
`using remove_all_extents_t = typename remove_all_extents< _Tp >::type`
- `template<typename _Tp >`
`using remove_const_t = typename remove_const< _Tp >::type`
- `template<typename _Tp >`
`using remove_cv_t = typename remove_cv< _Tp >::type`
- `template<typename _Tp >`
`using remove_extent_t = typename remove_extent< _Tp >::type`
- `template<typename _Tp >`
`using remove_pointer_t = typename remove_pointer< _Tp >::type`
- `template<typename _Tp >`
`using remove_reference_t = typename remove_reference< _Tp >::type`
- `template<typename _Tp >`
`using remove_volatile_t = typename remove_volatile< _Tp >::type`
- `template<typename _Tp >`
`using result_of_t = typename result_of< _Tp >::type`
- `typedef __SIZE_TYPE__ size_t`
- `typedef match_results< string::const_iterator > smatch`
- `typedef regex_iterator< string::const_iterator > sregex_iterator`
- `typedef regex_token_iterator< string::const_iterator > sregex_token_iterator`
- `typedef sub_match< string::const_iterator > ssub_match`
- `typedef basic_streambuf< char > streambuf`
- `typedef long long streamoff`
- `typedef fpos< mbstate_t > streampos`
- `typedef ptrdiff_t streamsize`
- `typedef basic_string< char > string`
- `using string_view = basic_string_view< char >`
- `typedef basic_stringbuf< char > stringbuf`
- `typedef basic_stringstream< char > stringstream`
- `typedef ratio< 1000000000000, 1 > tera`
- `typedef void(* terminate_handler) ()`
- `using true_type = integral_constant< bool, true >`
- `template<size_t __i, typename _Tp >`
`using tuple_element_t = typename tuple_element< __i, _Tp >::type`
- `typedef fpos< mbstate_t > u16streampos`
- `typedef basic_string< char16_t > u16string`
- `using u16string_view = basic_string_view< char16_t >`
- `typedef fpos< mbstate_t > u32streampos`
- `typedef basic_string< char32_t > u32string`
- `using u32string_view = basic_string_view< char32_t >`
- `template<typename _Tp >`
`using underlying_type_t = typename underlying_type< _Tp >::type`
- `typedef void(* unexpected_handler) ()`
- `template<size_t _Np, typename _Variant >`
`using variant_alternative_t = typename variant_alternative< _Np, _Variant >::type`
- `template<typename... >`
`using void_t = void`

- typedef [match_results](#)< const wchar_t * > **wcmatch**
- typedef [regex_iterator](#)< const wchar_t * > **wcregex_iterator**
- typedef [regex_token_iterator](#)< const wchar_t * > **wcregex_token_iterator**
- typedef [sub_match](#)< const wchar_t * > **wcsub_match**
- typedef [basic_filebuf](#)< wchar_t > **wfilebuf**
- typedef [basic_fstream](#)< wchar_t > **wfstream**
- typedef [basic_ifstream](#)< wchar_t > **wifstream**
- typedef [basic_ios](#)< wchar_t > **wios**
- typedef [basic_iostream](#)< wchar_t > **wiostream**
- typedef [basic_istream](#)< wchar_t > **wistream**
- typedef [basic_istreamstream](#)< wchar_t > **wistreamstream**
- typedef [basic_ofstream](#)< wchar_t > **wofstream**
- typedef [basic_ostream](#)< wchar_t > **wostream**
- typedef [basic_ostreamstream](#)< wchar_t > **wostreamstream**
- typedef [basic_regex](#)< wchar_t > **wregex**
- typedef [match_results](#)< wstring::const_iterator > **wsmatch**
- typedef [regex_iterator](#)< wstring::const_iterator > **wsregex_iterator**
- typedef [regex_token_iterator](#)< wstring::const_iterator > **wsregex_token_iterator**
- typedef [sub_match](#)< wstring::const_iterator > **wssub_match**
- typedef [basic_streambuf](#)< wchar_t > **wstreambuf**
- typedef [fpos](#)< mbstate_t > **wstreampos**
- typedef [basic_string](#)< wchar_t > **wstring**
- using **wstring_view** = [basic_string_view](#)< wchar_t >
- typedef [basic_stringbuf](#)< wchar_t > **wstringbuf**
- typedef [basic_stringstream](#)< wchar_t > **wstringstream**

- template<typename _Tp >
using [remove_cvref_t](#) = typename remove_cvref< _Tp >::type

- template<typename _Tp >
using [type_identity_t](#) = typename type_identity< _Tp >::type

- template<typename _Tp >
using **unwrap_reference_t** = typename [unwrap_reference](#)< _Tp >::type

- template<typename _Tp >
using **unwrap_ref_decay_t** = typename [unwrap_ref_decay](#)< _Tp >::type

Enumerations

- enum { **_S_threshold** }
- enum { **_S_chunk_size** }
- enum { **_S_word_bit** }
- enum class **__barrier_phase_t** : unsigned char
- enum **__memory_order_modifier** { **__memory_order_mask** , **__memory_order_modifier_mask** , **__memory_order_hle_acquire** , **__memory_order_hle_release** }

- enum `_ios_Fmtflags` {
`_S_boolalpha`, `_S_dec`, `_S_fixed`, `_S_hex`,
`_S_internal`, `_S_left`, `_S_oct`, `_S_right`,
`_S_scientific`, `_S_showbase`, `_S_showpoint`, `_S_showpos`,
`_S_skipws`, `_S_unitbuf`, `_S_uppercase`, `_S_adjustfield`,
`_S_basefield`, `_S_floatfield`, `_S_ios_fmtflags_end`, `_S_ios_fmtflags_max`,
`_S_ios_fmtflags_min` }
- enum `_ios_iostate` {
`_S_goodbit`, `_S_badbit`, `_S_eofbit`, `_S_failbit`,
`_S_ios_iostate_end`, `_S_ios_iostate_max`, `_S_ios_iostate_min` }
- enum `_ios_Openmode` {
`_S_app`, `_S_at`, `_S_bin`, `_S_in`,
`_S_out`, `_S_trunc`, `_S_noreplace`, `_S_ios_openmode_end`,
`_S_ios_openmode_max`, `_S_ios_openmode_min` }
- enum `_ios_Seekdir` { `_S_beg`, `_S_cur`, `_S_end`, `_S_ios_seekdir_end` }
- enum `_Manager_operation` { `__get_type_info`, `__get_func_ptr`, `__clone_func`, `__destroy_func` }
- enum `_Rb_tree_color` { `_S_red`, `_S_black` }
- enum class `align_val_t` : `size_t`
- enum class `byte` : unsigned char
- enum class `chars_format` { `scientific`, `fixed`, `hex`, `general` }
- enum `codecvt_mode` { `consume_header`, `generate_header`, `little_endian` }
- enum class `cv_status` { `no_timeout`, `timeout` }
- enum class `endian` { `little`, `big`, `native` }
- enum class `errc` {
`address_family_not_supported`, `address_in_use`, `address_not_available`, `already_connected`,
`argument_list_too_long`, `argument_out_of_domain`, `bad_address`, `bad_file_descriptor`,
`broken_pipe`, `connection_aborted`, `connection_already_in_progress`, `connection_refused`,
`connection_reset`, `cross_device_link`, `destination_address_required`, `device_or_resource_busy`,
`directory_not_empty`, `executable_format_error`, `file_exists`, `file_too_large`,
`filename_too_long`, `function_not_supported`, `host_unreachable`, `illegal_byte_sequence`,
`inappropriate_io_control_operation`, `interrupted`, `invalid_argument`, `invalid_seek`,
`io_error`, `is_a_directory`, `message_size`, `network_down`,
`network_reset`, `network_unreachable`, `no_buffer_space`, `no_child_process`,
`no_lock_available`, `no_message`, `no_protocol_option`, `no_space_on_device`,
`no_such_device_or_address`, `no_such_device`, `no_such_file_or_directory`, `no_such_process`,
`not_a_directory`, `not_a_socket`, `not_connected`, `not_enough_memory`,
`operation_in_progress`, `operation_not_permitted`, `operation_not_supported`, `operation_would_block`,
`permission_denied`, `protocol_not_supported`, `read_only_file_system`, `resource_deadlock_would_occur`,
`resource_unavailable_try_again`, `result_out_of_range`, `timed_out`, `too_many_files_open_in_system`,
`too_many_files_open`, `too_many_links`, `too_many_symbolic_link_levels`, `wrong_protocol_type` }
- enum `float_denorm_style` { `denorm_indeterminate`, `denorm_absent`, `denorm_present` }
- enum `float_round_style` {
`round_indeterminate`, `round_toward_zero`, `round_to_nearest`, `round_toward_infinity`,
`round_toward_neg_infinity` }
- enum class `future_errc` { `future_already_retrieved`, `promise_already_satisfied`, `no_state`, `broken_promise` }
- enum class `future_status` { `ready`, `timeout`, `deferred` }
- enum class `io_errc` { `stream` }
- enum class `launch` { `async`, `deferred` }
- enum class `memory_order` : int {
`relaxed`, `consume`, `acquire`, `release`,
`acq_rel`, `seq_cst` }
- enum class `pointer_safety` { `relaxed`, `preferred`, `strict` }

Functions

- `template<typename _CharT >`
`_CharT * __add_grouping (_CharT * __s, _CharT __sep, const char * __gbeg, size_t __gsize, const _CharT * __first, const _CharT * __last)`
- `template<typename _Tp >`
`constexpr _Tp * __addressof (_Tp & __r) noexcept`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator __adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`constexpr void __adjust_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __len, _Tp __value, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance >`
`constexpr void __advance (_BidirectionalIterator & __i, _Distance __n, bidirectional_iterator_tag)`
- `template<typename _InputIterator, typename _Distance >`
`constexpr void __advance (_InputIterator & __i, _Distance __n, input_iterator_tag)`
- `template<typename _OutputIterator, typename _Distance >`
`void __advance (_OutputIterator &, _Distance, output_iterator_tag)=delete`
- `template<typename _RandomAccessIterator, typename _Distance >`
`constexpr void __advance (_RandomAccessIterator & __i, _Distance __n, random_access_iterator_tag)`
- `template<typename _Alloc >`
`constexpr void __alloc_on_copy (_Alloc & __one, const _Alloc & __two)`
- `template<typename _Alloc >`
`constexpr _Alloc __alloc_on_copy (const _Alloc & __a)`
- `template<typename _Alloc >`
`constexpr void __alloc_on_move (_Alloc & __one, _Alloc & __two)`
- `template<typename _Alloc >`
`constexpr void __alloc_on_swap (_Alloc & __one, _Alloc & __two)`
- `template<typename _Tp, _Lock_policy _Lp = __default_lock_policy, typename _Alloc, typename... _Args >`
`__shared_ptr< _Tp, _Lp > __allocate_shared (const _Alloc & __a, _Args &&... __args)`
- `template<typename _Fn, typename _Tuple, size_t... _Idx >`
`constexpr decltype(auto) __apply_impl (_Fn && __f, _Tuple && __t, index_sequence< _Idx... >)`
- `template<typename _Tp >`
`void __atomic_notify_address (const _Tp * __addr, bool __all) noexcept`
- `void __atomic_notify_address_bare (const __detail::__platform_wait_t * __addr, bool __all) noexcept`
- `template<typename _Tp, typename _Pred >`
`void __atomic_wait_address (const _Tp * __addr, _Pred __pred) noexcept`
- `template<typename _Pred >`
`void __atomic_wait_address_bare (const __detail::__platform_wait_t * __addr, _Pred __pred) noexcept`
- `template<typename _Tp, typename _Pred, typename _Rep, typename _Period >`
`bool __atomic_wait_address_for (const _Tp * __addr, _Pred __pred, const chrono::duration< _Rep, _Period > & __rtime) noexcept`
- `template<typename _Pred, typename _Rep, typename _Period >`
`bool __atomic_wait_address_for_bare (const __detail::__platform_wait_t * __addr, _Pred __pred, const chrono::duration< _Rep, _Period > & __rtime) noexcept`
- `template<typename _Tp, typename _ValFn, typename _Rep, typename _Period >`
`bool __atomic_wait_address_for_v (const _Tp * __addr, _Tp && __old, _ValFn && __vfn, const chrono::duration< _Rep, _Period > & __rtime) noexcept`
- `template<typename _Tp, typename _Pred, typename _Clock, typename _Dur >`
`bool __atomic_wait_address_until (const _Tp * __addr, _Pred __pred, const chrono::time_point< _Clock, _Dur > & __atime) noexcept`

- `template<typename _Pred, typename _Clock, typename _Dur >`
`bool __atomic_wait_address_until_bare (const __detail::__platform_wait_t * __addr, _Pred __pred, const chrono::time_point< _Clock, _Dur > & __atime) noexcept`
- `template<typename _Tp, typename _ValFn, typename _Clock, typename _Dur >`
`bool __atomic_wait_address_until_v (const _Tp * __addr, _Tp && __old, _ValFn && __vfn, const chrono::time_point< _Clock, _Dur > & __atime) noexcept`
- `template<typename _Tp, typename _ValFn >`
`void __atomic_wait_address_v (const _Tp * __addr, _Tp __old, _ValFn __vfn) noexcept`
- `template<typename _Facet >`
`const _Facet & __check_facet (const _Facet * __f)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`constexpr void __chunk_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance __chunk_size, _Compare __comp)`
- `constexpr memory_order __cmpexch_failure_order (memory_order __m) noexcept`
- `constexpr memory_order __cmpexch_failure_order2 (memory_order __m) noexcept`
- `template<typename _Tp >`
`_Tp __complex_abs (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_acos (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_acosh (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`_Tp __complex_arg (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_asin (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_asinh (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_atan (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_atanh (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_cos (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_cosh (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_exp (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_log (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_pow (const complex< _Tp > & __x, const complex< _Tp > & __y)`
- `template<typename _Tp >`
`complex< _Tp > __complex_pow_unsigned (complex< _Tp > __x, unsigned __n)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_proj (const std::complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sin (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sinh (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sqrt (const complex< _Tp > & __z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_tan (const complex< _Tp > & __z)`

- `template<typename _Tp >`
`complex<_Tp> __complex_tanh (const complex<_Tp> &__z)`
- `int __convert_from_v (const __c_locale &__cloc, char *__out, const int __size, const char *__fmt,...)`
- `template<typename _Tp >`
`void __convert_to_v (const char *, _Tp &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void __convert_to_v (const char *, double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void __convert_to_v (const char *, float &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void __convert_to_v (const char *, long double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<bool _IsMove, typename _II, typename _OI >`
`constexpr _OI __copy_move_a (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _II, typename _Ite, typename _Seq, typename _Cat >`
`__gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> __copy_move_a (_II __first, _II __last, const __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> &)`
- `template<bool _IsMove, typename _IIte, typename _ISeq, typename _ICat, typename _OIte, typename _OSeq, typename _OCat >`
`__gnu_debug::Safe_iterator<_OIte, _OSeq, _OCat> __copy_move_a (const __gnu_debug::Safe_iterator<_IIte, _ISeq, _ICat> &, const __gnu_debug::Safe_iterator<_IIte, _ISeq, _ICat> &, const __gnu_debug::Safe_iterator<_OIte, _OSeq, _OCat> &)`
- `template<bool _IsMove, typename _Ite, typename _Seq, typename _Cat, typename _OI >`
`_OI __copy_move_a (const __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> &, const __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> &, _OI)`
- `template<bool _IsMove, typename _ITp, typename _IRef, typename _IPtr, typename _OTp >`
`::Deque_iterator<_OTp, _OTp &, _OTp*> __copy_move_a1 (::Deque_iterator<_ITp, _IRef, _IPtr> __first, ::Deque_iterator<_ITp, _IRef, _IPtr> __last, ::Deque_iterator<_OTp, _OTp &, _OTp*> __result)`
- `template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >`
`_OI __copy_move_a1 (::Deque_iterator<_Tp, _Ref, _Ptr> __first, ::Deque_iterator<_Tp, _Ref, _Ptr> __last, _OI __result)`
- `template<bool _IsMove, typename _II, typename _Tp >`
`__gnu_cxx::enable_if<__is_random_access_iter<_II>::value, ::Deque_iterator<_Tp, _Tp &, _Tp*>>::type __copy_move_a1 (_II __first, _II __last, ::Deque_iterator<_Tp, _Tp &, _Tp*> __result)`
- `template<bool _IsMove, typename _II, typename _OI >`
`constexpr _OI __copy_move_a1 (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, ostreambuf_iterator<_CharT, char_traits<_CharT>>>::type __copy_move_a2 (_CharT *, _CharT *, ostreambuf_iterator<_CharT, char_traits<_CharT>> __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, ostreambuf_iterator<_CharT>>::type __copy_move_a2 (_CharT * __first, _CharT * __last, ostreambuf_iterator<_CharT> __result)`
- `template<bool _IsMove, typename _II, typename _OI >`
`constexpr _OI __copy_move_a2 (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, ostreambuf_iterator<_CharT, char_traits<_CharT>>>::type __copy_move_a2 (const _CharT *, const _CharT *, ostreambuf_iterator<_CharT, char_traits<_CharT>> __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, ostreambuf_iterator<_CharT>>::type __copy_move_a2 (const _CharT * __first, const _CharT * __last, ostreambuf_iterator<_CharT> __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, _CharT*>::type __copy_move_a2 (istreambuf_iterator<_CharT> __first, istreambuf_iterator<_CharT> __last, _CharT * __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::enable_if<__is_char<_CharT>::value, ::Deque_iterator<_CharT, _CharT &, _CharT*>::type __copy_move_a2 (istreambuf_iterator<_CharT, char_traits<_CharT>> __first,`

```

istreambuf_iterator<_CharT, char_traits<_CharT>> __last, ::_Deque_iterator<_CharT, _CharT &, _CharT *
> __result)
• template<bool _IsMove, typename _CharT >
    __gnu_cxx::__enable_if<__is_char<_CharT>::__value, _CharT * >::__type __copy_move_a2(istreambuf_iterator<
    _CharT, char_traits<_CharT>>, istreambuf_iterator<_CharT, char_traits<_CharT>>, _CharT *)
• template<bool _IsMove, typename _II, typename _OI >
    constexpr _OI __copy_move_backward_a(_II __first, _II __last, _OI __result)
• template<bool _IsMove, typename _II, typename _Ite, typename _Seq, typename _Cat >
    __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> __copy_move_backward_a(_II __first, _II __last, const __gnu_debug::Safe_iterator<
    _Ite, _Seq, _Cat> &)
• template<bool _IsMove, typename _IIte, typename _ISeq, typename _ICat, typename _OIte, typename _OSeq, typename _OCat >
    __gnu_debug::Safe_iterator<_OIte, _OSeq, _OCat> __copy_move_backward_a(const __gnu_debug::Safe_iterator<
    _IIte, _ISeq, _ICat> &, const __gnu_debug::Safe_iterator<_IIte, _ISeq, _ICat> &, const __gnu_debug::Safe_iterator<
    _OIte, _OSeq, _OCat> &)
• template<bool _IsMove, typename _Ite, typename _Seq, typename _Cat, typename _OI >
    _OI __copy_move_backward_a(const __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> &, const
    __gnu_debug::Safe_iterator<_Ite, _Seq, _Cat> &, _OI)
• template<bool _IsMove, typename _ITp, typename _IRef, typename _IPtr, typename _OTp >
    ::_Deque_iterator<_OTp, _OTp &, _OTp * > __copy_move_backward_a1(::_Deque_iterator<_ITp, _IRef,
    _IPtr> __first, ::_Deque_iterator<_ITp, _IRef, _IPtr> __last, ::_Deque_iterator<_OTp, _OTp &, _OTp * >
    __result)
• template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
    _OI __copy_move_backward_a1(::_Deque_iterator<_Tp, _Ref, _Ptr> __first, ::_Deque_iterator<_Tp, _Ref,
    _Ptr> __last, _OI __result)
• template<bool _IsMove, typename _BI1, typename _BI2 >
    constexpr _BI2 __copy_move_backward_a1(_BI1 __first, _BI1 __last, _BI2 __result)
• template<bool _IsMove, typename _II, typename _Tp >
    __gnu_cxx::__enable_if<__is_random_access_iter<_II>::__value, ::_Deque_iterator<_Tp, _Tp &, _Tp * >
    >::__type __copy_move_backward_a1(_II __first, _II __last, ::_Deque_iterator<_Tp, _Tp &, _Tp * > __result)
• template<bool _IsMove, typename _BI1, typename _BI2 >
    constexpr _BI2 __copy_move_backward_a2(_BI1 __first, _BI1 __last, _BI2 __result)
• template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
    _OI __copy_move_backward_dit(::_Deque_iterator<_Tp, _Ref, _Ptr> __first, ::_Deque_iterator<_Tp, _Ref,
    _Ptr> __last, _OI __result)
• template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
    _OI __copy_move_dit(::_Deque_iterator<_Tp, _Ref, _Ptr> __first, ::_Deque_iterator<_Tp, _Ref, _Ptr>
    __last, _OI __result)
• template<typename _InputIterator, typename _Size, typename _OutputIterator >
    constexpr _OutputIterator __copy_n(_InputIterator __first, _Size __n, _OutputIterator __result, input_iterator_tag)
• template<typename _RandomAccessIterator, typename _Size, typename _OutputIterator >
    constexpr _OutputIterator __copy_n(_RandomAccessIterator __first, _Size __n, _OutputIterator __result,
    random_access_iterator_tag)
• template<typename _InputIterator, typename _Size, typename _OutputIterator >
    constexpr _OutputIterator __copy_n_a(_InputIterator __first, _Size __n, _OutputIterator __result, bool)
• template<typename _CharT, typename _Size >
    __gnu_cxx::__enable_if<__is_char<_CharT>::__value, _CharT * >::__type __copy_n_a(istreambuf_iterator<
    _CharT> __it, _Size __n, _CharT * __result, bool __strict)
• template<typename _CharT, typename _Size >
    __gnu_cxx::__enable_if<__is_char<_CharT>::__value, ::_Deque_iterator<_CharT, _CharT &, _CharT * >
    >::__type __copy_n_a(istreambuf_iterator<_CharT, char_traits<_CharT>> __it, _Size __size, ::_Deque_iterator<
    _CharT, _CharT &, _CharT * > __result, bool __strict)

```

- `template<typename _CharT, typename _Size >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type __copy_n_a (istreambuf_iterator<`
`_CharT, char_traits< _CharT > >, _Size, _CharT *, bool)`
- `template<typename _CharT, typename _Traits >`
`streamsize __copy_streambufs (basic_streambuf< _CharT, _Traits > * __sbin, basic_streambuf< _CharT, _`
`Traits > * __sout)`
- `template<typename _CharT, typename _Traits >`
`streamsize __copy_streambufs_eof (basic_streambuf< _CharT, _Traits > *, basic_streambuf< _CharT, _Traits`
`> *, bool &)`
- `template<> streamsize __copy_streambufs_eof (basic_streambuf< char > * __sbin, basic_streambuf< char`
`> * __sout, bool & __ineof)`
- `template<> streamsize __copy_streambufs_eof (basic_streambuf< wchar_t > * __sbin, basic_streambuf<`
`wchar_t > * __sout, bool & __ineof)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr iterator_traits< _InputIterator >::difference_type __count_if (_InputIterator __first, _InputIterator __`
`last, _Predicate __pred)`
- `constexpr size_t __deque_buf_size (size_t __size)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::difference_type __distance (_InputIterator __first, _InputIterator __`
`last, input_iterator_tag)`
- `template<typename _OutputIterator >`
`void __distance (_OutputIterator, _OutputIterator, output_iterator_tag)=delete`
- `template<typename _RandomAccessIterator >`
`constexpr iterator_traits< _RandomAccessIterator >::difference_type __distance (_RandomAccessIterator __`
`first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >`
`bool __do_str_codecvt (const _InChar * __first, const _InChar * __last, _OutStr & __outstr, const _Codecvt & __`
`cvt, _State & __state, size_t & __count, _Fn __fn)`
- `template<typename _Result_type, typename _Visitor, typename... _Variants>`
`constexpr decltype(auto) __do_visit (_Visitor && __visitor, _Variants &&... __variants)`
- `template<typename _I1, typename _I2 >`
`constexpr bool __equal4 (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _I2, typename _BinaryPredicate >`
`constexpr bool __equal4 (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _I1, typename _I2 >`
`constexpr bool __equal_aux (_I1 __first1, _I1 __last1, _I2 __first2)`
- `template<typename _I1, typename _I2, typename _Seq2, typename _Cat2 >`
`bool __equal_aux (_I1, _I1, const ::__gnu_debug::__Safe_iterator< _I2, _Seq2, _Cat2 > &)`
- `template<typename _I1, typename _Seq1, typename _Cat1, typename _I2 >`
`bool __equal_aux (const ::__gnu_debug::__Safe_iterator< _I1, _Seq1, _Cat1 > &, const ::__gnu_debug::__Safe_iterator<`
`_I1, _Seq1, _Cat1 > &, _I2)`
- `template<typename _I1, typename _Seq1, typename _Cat1, typename _I2, typename _Seq2, typename _Cat2 >`
`bool __equal_aux (const ::__gnu_debug::__Safe_iterator< _I1, _Seq1, _Cat1 > &, const ::__gnu_debug::__Safe_iterator<`
`_I1, _Seq1, _Cat1 > &, const ::__gnu_debug::__Safe_iterator< _I2, _Seq2, _Cat2 > &)`
- `template<typename _Tp, typename _Ref, typename _Ptr, typename _II >`
`__gnu_cxx::__enable_if< __is_random_access_iter< _II >::__value, bool >::__type __equal_aux1 (::__`
`Deque_iterator< _Tp, _Ref, _Ptr > __first1, ::Deque_iterator< _Tp, _Ref, _Ptr > __last1, _II __first2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool __equal_aux1 (::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, ::Deque_iterator< _Tp1, _Ref1, _Ptr1`
`> __last1, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2)`
- `template<typename _II, typename _Tp, typename _Ref, typename _Ptr >`
`__gnu_cxx::__enable_if< __is_random_access_iter< _II >::__value, bool >::__type __equal_aux1 (_II __first1,`
`_II __last1, ::Deque_iterator< _Tp, _Ref, _Ptr > __first2)`

- `template<typename _II1, typename _II2 >`
`constexpr bool __equal_aux1 (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _Tp, typename _Ref, typename _Ptr, typename _II >`
`bool __equal_dit (const ::_Deque_iterator< _Tp, _Ref, _Ptr > &__first1, const ::_Deque_iterator< _Tp, _Ref, _Ptr > &__last1, _II __first2)`
- `template<typename _ForwardIterator, typename _Tp, typename _CompareItTp, typename _CompareTplt >`
`constexpr pair< _ForwardIterator, _ForwardIterator > __equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _CompareItTp __comp_it_val, _CompareTplt __comp_val_it)`
- `template<typename _Tp, typename _Up = _Tp>`
`constexpr _Tp __exchange (_Tp &__obj, _Up &&__new_val)`
- `template<typename _Flte, typename _Tp >`
`constexpr void __fill_a (_Flte __first, _Flte __last, const _Tp &__value)`
- `template<typename _Ite, typename _Seq, typename _Cat, typename _Tp >`
`void __fill_a (const ::_gnu_debug::_Safe_iterator< _Ite, _Seq, _Cat > &, const ::_gnu_debug::_Safe_iterator< _Ite, _Seq, _Cat > &, const _Tp &)`
- `template<typename _Ite, typename _Cont, typename _Tp >`
`constexpr void __fill_a1 (::_gnu_cxx::_normal_iterator< _Ite, _Cont > __first, ::_gnu_cxx::_normal_iterator< _Ite, _Cont > __last, const _Tp &__value)`
- `constexpr void __fill_a1 (::_Bit_iterator, ::_Bit_iterator, const bool &)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _gnu_cxx::_enable_if<!__is_scalar< _Tp >::__value, void >::__type __fill_a1 (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _gnu_cxx::_enable_if< __is_scalar< _Tp >::__value, void >::__type __fill_a1 (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _Tp >`
`constexpr _gnu_cxx::_enable_if< __is_byte< _Tp >::__value, void >::__type __fill_a1 (_Tp *__first, _Tp *__last, const _Tp &__c)`
- `template<typename _Tp, typename _VTp >`
`void __fill_a1 (const ::_Deque_iterator< _Tp, _Tp &, _Tp * > &__first, const ::_Deque_iterator< _Tp, _Tp &, _Tp * > &__last, const _VTp &__value)`
- `constexpr void __fill_bvector (_Bit_type *__v, unsigned int __first, unsigned int __last, bool __x) noexcept`
- `constexpr void __fill_bvector_n (_Bit_type *, size_t, bool) noexcept`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator __fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::input_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator __fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::output_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator __fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::random_access_iterator_tag)`
- `template<typename _Ite, typename _Seq, typename _Cat, typename _Size, typename _Tp >`
`::_gnu_debug::_Safe_iterator< _Ite, _Seq, _Cat > __fill_n_a (const ::_gnu_debug::_Safe_iterator< _Ite, _Seq, _Cat > &__first, _Size __n, const _Tp &__value, std::input_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _gnu_cxx::_enable_if<!__is_scalar< _Tp >::__value, _OutputIterator >::__type __fill_n_a1 (_OutputIterator __first, _Size __n, const _Tp &__value)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _gnu_cxx::_enable_if< __is_scalar< _Tp >::__value, _OutputIterator >::__type __fill_n_a1 (_OutputIterator __first, _Size __n, const _Tp &__value)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __final_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BinaryPredicate >`
`constexpr _BidirectionalIterator1 find_end (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, ↵`
`_BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, bidirectional_iterator_tag, bidirectional_iterator_tag,`
`_BinaryPredicate __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _Forward↵`
`Iterator2 __first2, _ForwardIterator2 __last2, forward_iterator_tag, forward_iterator_tag, _BinaryPredicate __↵`
`comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, input_iterator_tag)`
- `template<typename _Iterator, typename _Predicate >`
`constexpr _Iterator find_if (_Iterator __first, _Iterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator, typename _Predicate >`
`constexpr _RandomAccessIterator find_if (_RandomAccessIterator __first, _RandomAccessIterator __last, ↵`
`_Predicate __pred, random_access_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate, typename _Distance >`
`constexpr _InputIterator find_if_not_n (_InputIterator __first, _Distance &__len, _Predicate __pred)`
- `template<typename _Tp, typename... _Types>`
`constexpr size_t find_uniq_type_in_pack ()`
- `template<typename _EuclideanRingElement >`
`constexpr _EuclideanRingElement gcd (_EuclideanRingElement __m, _EuclideanRingElement __n)`
- `template<typename _IntType, typename _UniformRandomBitGenerator >`
`pair< _IntType, _IntType > gen_two_uniform_ints (_IntType __b0, _IntType __b1, _UniformRandomBit↵`
`Generator &&__g)`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & get_helper (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & get_helper (const Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<size_t __i, typename... _Types>`
`__enable_if_t<(__i >= sizeof...(_Types))> get_helper (const tuple< _Types... > &)=delete`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void heap_select (_RandomAccessIterator __first, _RandomAccessIterator __middle, _Random↵`
`AccessIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`_Tp hypot3 (_Tp __x, _Tp __y, _Tp __z)`
- `template<typename _Tp >`
`size_t iconv_adaptor (size_t(*__func)(iconv_t, _Tp, size_t *, char **, size_t *), iconv_t __cd, char **__inbuf,`
`size_t *__inbytes, char **__outbuf, size_t *__outbytes)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`constexpr bool includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵`
`Iterator2 __last2, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last,`
`_Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void inplace_stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare ↵`
`__comp)`
- `template<typename _CharT, typename _ValueT >`
`int int_to_char (_CharT *__bufend, _ValueT __v, const _CharT *__lit, ios_base::fmtflags __flags, bool __dec)`

- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`constexpr void __introspect (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`constexpr void __introsort_loop (_RandomAccessIterator __first, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _Tp, typename _Up = typename __inv_unwrap<_Tp>::type>`
`constexpr _Up && __invfwd (typename remove_reference<_Tp>::type &__t) noexcept`
- `template<typename _Callable, typename... _Args>`
`constexpr __invoke_result<_Callable, _Args...>::type __invoke (_Callable &&__fn, _Args &&... __args)`
`noexcept(__is_nothrow_invocable<_Callable, _Args...>::value)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`constexpr _Res __invoke_impl (__invoke_memfun_deref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`constexpr _Res __invoke_impl (__invoke_memfun_ref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`constexpr _Res __invoke_impl (__invoke_memobj_deref, _MemPtr &&__f, _Tp &&__t)`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`constexpr _Res __invoke_impl (__invoke_memobj_ref, _MemPtr &&__f, _Tp &&__t)`
- `template<typename _Res, typename _Fn, typename... _Args>`
`constexpr _Res __invoke_impl (__invoke_other, _Fn &&__f, _Args &&... __args)`
- `template<typename _Res, typename _Callable, typename... _Args>`
`constexpr enable_if_t<__is_invocable_r_v<_Res, _Callable, _Args...>, _Res> __invoke_r (_Callable &&__fn, _Args &&... __args) noexcept(__is_nothrow_invocable_r_v<_Res, _Callable, _Args...>)`
- `constexpr bool __is_constant_evaluated () noexcept`
- `template<typename _RandomAccessIterator, typename _Compare, typename _Distance >`
`constexpr bool __is_heap (_RandomAccessIterator __first, _Compare __comp, _Distance __n)`
- `template<typename _RandomAccessIterator, typename _Distance >`
`constexpr bool __is_heap (_RandomAccessIterator __first, _Distance __n)`
- `template<typename _RandomAccessIterator >`
`constexpr bool __is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr bool __is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`constexpr _Distance __is_heap_until (_RandomAccessIterator __first, _Distance __n, _Compare &__comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool __is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool __is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator __is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `constexpr bool __is_valid_cmpexch_failure_order (memory_order __m) noexcept`
- `template<typename _CharT, typename _Traits >`
`void __istream_extract (basic_istream<_CharT, _Traits> &, _CharT *, streamsize)`
- `void __istream_extract (istream &, char *, streamsize)`
- `template<typename _Iter >`
`constexpr iterator_traits<_Iter>::iterator_category __iterator_category (const _Iter &)`
- `template<typename _Fp >`
`constexpr _Fp __lerp (_Fp __a, _Fp __b, _Fp __t) noexcept`

- `template<typename _Tp1, typename _Ref, typename _Ptr, typename _Tp2 >`
`int __lex_cmp_dit (::Deque_iterator< _Tp1, _Ref, _Ptr > __first1, ::Deque_iterator< _Tp1, _Ref, _Ptr > __last1, const _Tp2 *__first2, const _Tp2 *__last2)`
- `template<typename _II1, typename _II2 >`
`constexpr bool __lexicographical_compare_aux (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _Ite2, typename _Seq2, typename _Cat2 >`
`bool __lexicographical_compare_aux (_II1 __first1, _II1 __last1, const ::gnu_debug::Safe_iterator< _Ite2, _Seq2, _Cat2 > & __first2, const ::gnu_debug::Safe_iterator< _Ite2, _Seq2, _Cat2 > & __last2)`
- `template<typename _II1, typename _Iter2, typename _Seq2, typename _Cat2 >`
`bool __lexicographical_compare_aux (_II1, _II1, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &)`
- `template<typename _Ite1, typename _Seq1, typename _Cat1, typename _II2 >`
`bool __lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __first1, const ::gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __last1, _II2 __first2, _II2 __last2)`
- `template<typename _Ite1, typename _Seq1, typename _Cat1, typename _Ite2, typename _Seq2, typename _Cat2 >`
`bool __lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __first1, const ::gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __last1, const ::gnu_debug::Safe_iterator< _Ite2, _Seq2, _Cat2 > & __first2, const ::gnu_debug::Safe_iterator< _Ite2, _Seq2, _Cat2 > & __last2)`
- `template<typename _Iter1, typename _Seq1, typename _Cat1, typename _II2 >`
`bool __lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, _II2, _II2)`
- `template<typename _Iter1, typename _Seq1, typename _Cat1, typename _Iter2, typename _Seq2, typename _Cat2 >`
`bool __lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool __lexicographical_compare_aux1 (::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, ::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2 >`
`bool __lexicographical_compare_aux1 (::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, ::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, _Tp2 *__first2, _Tp2 *__last2)`
- `template<typename _II1, typename _II2 >`
`constexpr bool __lexicographical_compare_aux1 (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _Tp1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool __lexicographical_compare_aux1 (_Tp1 *__first1, _Tp1 *__last1, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`constexpr bool __lexicographical_compare_impl (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, __gnu_cxx::Compare __comp)`
- `constexpr int __lg (int __n)`
- `constexpr long __lg (long __n)`
- `constexpr long long __lg (long long __n)`
- `constexpr unsigned __lg (unsigned __n)`
- `constexpr unsigned long __lg (unsigned long __n)`
- `constexpr unsigned long long __lg (unsigned long long __n)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator __lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)`
- `template<typename _Tp, typename _Tuple, size_t... _Idx>`
`constexpr _Tp __make_from_tuple_impl (_Tuple && __t, index_sequence< _Idx... >)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare &__comp)`
- `template<typename _Iterator, typename _ReturnType = __conditional_t<__move_if_noexcept_cond <typename iterator_traits<_Iterator>::value_type>::value, _Iterator, move_iterator<_Iterator>>>>`
`constexpr _ReturnType __make_move_if_noexcept_iterator (_Iterator __i)`
- `template<typename _Tp, typename _ReturnType = __conditional_t<__move_if_noexcept_cond<_Tp>::value, const _Tp*, move_iterator<_Tp*>>>>`
`constexpr _ReturnType __make_move_if_noexcept_iterator (_Tp * __i)`
- `template<typename _Iterator >`
`constexpr reverse_iterator<_Iterator> __make_reverse_iterator (_Iterator __i)`
- `template<typename _Tp, _Lock_policy _Lp = __default_lock_policy, typename... _Args>`
`__shared_ptr<_Tp, _Lp> __make_shared (_Args &&... __args)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator __max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp, typename _Up >`
`constexpr int __memcmp (const _Tp * __first1, const _Up * __first2, size_t __num)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator __merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Pointer, typename _Compare >`
`void __merge_adaptive (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator1, typename _RandomAccessIterator2, typename _Distance, typename _Compare >`
`void __merge_sort_loop (_RandomAccessIterator1 __first, _RandomAccessIterator1 __last, _RandomAccessIterator2 __result, _Distance __step_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Compare >`
`void __merge_sort_with_buffer (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Compare >`
`void __merge_without_buffer (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator __min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr pair<_ForwardIterator, _ForwardIterator> __minmax_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> __mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> __mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _Iterator >`
`constexpr _Iterator __miter_base (_Iterator __it)`
- `template<typename _Iterator >`
`constexpr auto __miter_base (move_iterator<_Iterator> __it) -> decltype(__miter_base(__it.base()))`
- `template<typename _Iterator >`
`constexpr auto __miter_base (reverse_iterator<_Iterator> __it) -> decltype(__make_reverse_iterator(__miter_base(__it.base())))`

- `template<typename _Iterator, typename _Compare >`
`constexpr void __move_median_to_first (_Iterator __result, _Iterator __a, _Iterator __b, _Iterator __c, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Compare >`
`_OutputIterator __move_merge (_InputIterator __first1, _InputIterator __last1, _InputIterator __first2, _InputIterator __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`void __move_merge_adaptive (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BidirectionalIterator3, typename _Compare >`
`void __move_merge_adaptive_backward (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, _BidirectionalIterator3 __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool __next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Iterator, typename _Container >`
`constexpr _Iterator __niter_base (_gnu_cxx::__normal_iterator<_Iterator, _Container > __it) noexcept(/*conditional */)`
- `template<typename _Iterator >`
`constexpr _Iterator __niter_base (_Iterator __it) noexcept(/*conditional */)`
- `template<typename _Ite, typename _Seq >`
`_Ite __niter_base (const ::gnu_debug::Safe_iterator<_Ite, _Seq, std::random_access_iterator_tag > &)`
- `template<typename _Iterator >`
`constexpr auto __niter_base (move_iterator<_Iterator > __it) -> decltype(make_move_iterator(__niter_base(__it.base())))`
- `template<typename _Iterator >`
`constexpr auto __niter_base (reverse_iterator<_Iterator > __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- `template<typename _From, typename _To >`
`constexpr _From __niter_wrap (_From __from, _To __res)`
- `template<typename _Iterator >`
`constexpr _Iterator __niter_wrap (const _Iterator &, _Iterator __res)`
- `void __once_proxy (void)`
- `template<typename _CharT, typename _Traits >`
`void __ostream_fill (basic_ostream<_CharT, _Traits > &__out, streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream<_CharT, _Traits > & __ostream_insert (basic_ostream<_CharT, _Traits > &__out, const _CharT *__s, streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`void __ostream_write (basic_ostream<_CharT, _Traits > &__out, const _CharT *__s, streamsize __n)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator __partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Predicate >`
`constexpr _BidirectionalIterator __partition (_BidirectionalIterator __first, _BidirectionalIterator __last, _Predicate __pred, bidirectional_iterator_tag)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator __partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, forward_iterator_tag)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __result, _Compare &__comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool __prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`constexpr void __push_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __topIndex, _Tp __value, _Compare &__comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator __remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator __remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`constexpr _OutputIterator __replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _BidirectionalIterator >`
`constexpr void __reverse (_BidirectionalIterator __first, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`constexpr void __reverse (_RandomAccessIterator __first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator >`
`constexpr _BidirectionalIterator __rotate (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator __rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, forward_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator __rotate (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _Distance >`
`_BidirectionalIterator1 __rotate_adaptive (_BidirectionalIterator1 __first, _BidirectionalIterator1 __middle, _BidirectionalIterator1 __last, _Distance __len1, _Distance __len2, _BidirectionalIterator2 __buffer, _Distance __buffer_size)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Cat, typename _Size, typename _UniformRandomBitGenerator >`
`_OutputIterator __sample (_ForwardIterator __first, _ForwardIterator __last, forward_iterator_tag, _OutputIterator __out, _Cat, _Size __n, _UniformRandomBitGenerator &&__g)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Size, typename _UniformRandomBitGenerator >`
`_RandomAccessIterator __sample (_InputIterator __first, _InputIterator __last, input_iterator_tag, _RandomAccessIterator __out, random_access_iterator_tag, _Size __n, _UniformRandomBitGenerator &&__g)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 __search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`constexpr _ForwardIterator __search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`constexpr _ForwardIterator __search_n_aux (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred, std::forward_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Integer, typename _UnaryPredicate >`
`constexpr _RandomAccessIter __search_n_aux (_RandomAccessIter __first, _RandomAccessIter __last, _Integer __count, _UnaryPredicate __unary_pred, std::random_access_iterator_tag)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator __set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator __set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator __set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator __set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `constexpr long long __size_to_integer (double __n)`
- `constexpr long long __size_to_integer (float __n)`
- `constexpr int __size_to_integer (int __n)`
- `constexpr long __size_to_integer (long __n)`
- `constexpr long long __size_to_integer (long double __n)`
- `constexpr long long __size_to_integer (long long __n)`
- `constexpr unsigned __size_to_integer (unsigned __n)`
- `constexpr unsigned long __size_to_integer (unsigned long __n)`
- `constexpr unsigned long long __size_to_integer (unsigned long long __n)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare & __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator __stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Pointer, typename _Predicate, typename _Distance >`
`_ForwardIterator __stable_partition_adaptive (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, _Distance __len, _Pointer __buffer, _Distance __buffer_size)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Distance, typename _Compare >`
`void __stable_sort_adaptive (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_in (const char * __first, const char * __last, basic_string< _CharT, _Traits, _Alloc > & __outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_in (const char * __first, const char * __last, basic_string< _CharT, _Traits, _Alloc > & __outstr, const codecvt< _CharT, char, _State > & __cvt, _State & __state, size_t & __count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_in_all (const char * __first, const char * __last, basic_string< _CharT, _Traits, _Alloc > & __outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_out (const _CharT * __first, const _CharT * __last, basic_string< char, _Traits, _Alloc > & __outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_out (const _CharT * __first, const _CharT * __last, basic_string< char, _Traits, _Alloc > & __outstr, const codecvt< _CharT, char, _State > & __cvt, _State & __state, size_t & __count)`

- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool __str_codecvt_out_all (const _CharT * __first, const _CharT * __last, basic_string< char, _Traits, _Alloc >`
`& __outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `constexpr size_t __sv_check (size_t __size, size_t __pos, const char * __s)`
- `constexpr size_t __sv_limit (size_t __size, size_t __pos, size_t __off) noexcept`
- `void __terminate () noexcept`
- `void __throw_bad_alloc (void)`
- `void __throw_bad_any_cast ()`
- `void __throw_bad_array_new_length (void)`
- `void __throw_bad_cast (void)`
- `void __throw_bad_exception (void)`
- `void __throw_bad_function_call ()`
- `void __throw_bad_optional_access ()`
- `void __throw_bad_typeid (void)`
- `void __throw_bad_variant_access (bool __valueless)`
- `void __throw_bad_variant_access (const char * __what)`
- `void __throw_bad_weak_ptr ()`
- `void __throw_domain_error (const char *)`
- `void __throw_future_error (int)`
- `void __throw_invalid_argument (const char *)`
- `void __throw_ios_failure (const char *)`
- `void __throw_ios_failure (const char *, int)`
- `void __throw_length_error (const char *)`
- `void __throw_logic_error (const char *)`
- `void __throw_out_of_range (const char *)`
- `void __throw_out_of_range_fmt (const char *,...)`
- `void __throw_overflow_error (const char *)`
- `void __throw_range_error (const char *)`
- `void __throw_runtime_error (const char *)`
- `void __throw_system_error (int)`
- `void __throw_underflow_error (const char *)`
- `template<typename _Tp >`
`constexpr _Tp * __to_address (_Tp * __ptr) noexcept`
- `template<typename _Ptr >`
`constexpr auto __to_address (const _Ptr & __ptr) noexcept -> decltype(std::pointer_traits< _Ptr >::to_address(↵`
`__ptr))`
- `template<typename _Ptr, typename... _None>`
`constexpr auto __to_address (const _Ptr & __ptr, _None...) noexcept`
- `template<bool _Move = false, typename _Tp, size_t... _Idx>`
`constexpr array< remove_cv_t< _Tp >, sizeof...(_Idx)> __to_array (_Tp(& __a)[sizeof...(_Idx)], index_sequence<`
`_Idx... >)`
- `template<typename _Tp >`
`__detail::__integer_to_chars_result_type< _Tp > __to_chars_i (char * __first, char * __last, _Tp __value, int`
`__base=10)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __unguarded_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, ↵`
`_Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void __unguarded_linear_insert (_RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator __unguarded_partition (_RandomAccessIterator __first, _RandomAccess↵`
`Iterator __last, _RandomAccessIterator __pivot, _Compare __comp)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator __unguarded_partition_pivot (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Pointer, typename _ForwardIterator >`
`void __uninitialized_construct_buf (_Pointer __first, _Pointer __last, _ForwardIterator __seed)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator __unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator __unique_copy (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred, forward_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator __unique_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _BinaryPredicate __binary_pred, input_iterator_tag, forward_iterator_tag)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator __unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred, input_iterator_tag, output_iterator_tag)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator __upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< bool > __m, size_t __n, _Array< _Tp > __b, _Array< bool > __k)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s1, _Array< _Tp > __b, size_t __s2)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __e, _Array< size_t > __f, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __src, size_t __n, _Array< size_t > __i, _Array< _Tp > __dst, _Array< size_t > __j)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`

- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, size_t __s)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, _Tp *__restrict __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b, const size_t *__restrict __i)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __src, size_t __n, const size_t *__restrict __i, _Tp *__restrict __dst, const size_t *__restrict __j)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __src, size_t __n, size_t __s1, _Tp *__restrict __dst, size_t __s2)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy_construct (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__b, const _Tp *__e, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __o, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void __valarray_default_construct (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void __valarray_destroy_elements (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, _Array< size_t > __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, _Array< bool > __m, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, const size_t *__restrict __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, size_t __n, size_t __s, const _Tp &__t)`

- `template<typename _Tp >`
`void __valarray_fill_construct (_Tp *__b, _Tp *__e, const _Tp __t)`
- `template<typename _Tp >`
`_Tp * __valarray_get_storage (size_t)`
- `template<typename _Ta >`
`_Ta::value_type __valarray_max (const _Ta &__a)`
- `template<typename _Ta >`
`_Ta::value_type __valarray_min (const _Ta &__a)`
- `void __valarray_release_memory (void *__p)`
- `template<typename _Tp >`
`_Tp __valarray_sum (const _Tp *__f, const _Tp *__l)`
- `template<template< typename > class _Trait, typename _Tp, typename _Up = _Tp>`
`constexpr _Up __value_or (_Up __def=_Up()) noexcept`
- `template<typename... _Types, typename _Tp >`
`constexpr decltype(auto) __variant_cast (_Tp &&__rhs)`
- `bool __verify_grouping (const char *__grouping, size_t __grouping_size, const string &__grouping_tmp) throw`
`()`
- `template<typename _CharT, typename _Outlter >`
`_Outlter __write (_Outlter __s, const _CharT *__ws, int __len)`
- `template<typename _CharT >`
`ostreambuf_iterator< _CharT > __write (ostreambuf_iterator< _CharT > __s, const _CharT *__ws, int __len)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< bool > __m, const Expr< _Dom, _Tp`
`> &__e, size_t __n)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, const Expr< _Dom, _Tp`
`> &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool`
`> __m)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t`
`> __i)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __s, const Expr< _Dom, _Tp > &__e,`
`size_t __n)`

- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`

- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__divides (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`

- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`

- `template<typename _Tp, class _Dom >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool >`
`__m)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t >`
`__i)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___multiplies (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___plus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___plus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___plus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented___plus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`

- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_left (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`

- `template<typename _Tp >`
`void _Array_augmented_shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented_shift_right (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented_shift_right (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_shift_right (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, typename... _Args>`
`constexpr void _Construct (_Tp *__p, _Args &&... __args)`
- `template<typename _T1 >`
`void _Construct_novalue (_T1 *__p)`
- `template<typename _ForwardIterator >`
`constexpr void _Destroy (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Allocator >`
`constexpr void _Destroy (_ForwardIterator __first, _ForwardIterator __last, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void _Destroy (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp > &)`
- `template<typename _Tp >`
`constexpr void _Destroy (_Tp *__pointer)`
- `template<typename _ForwardIterator, typename _Size >`
`constexpr _ForwardIterator _Destroy_n (_ForwardIterator __first, _Size __count)`
- `size_t _Fnv_hash_bytes (const void *__ptr, size_t __len, size_t __seed)`
- `size_t _Hash_bytes (const void *__ptr, size_t __len, size_t __seed)`
- `unsigned int _Rb_tree_black_count (const _Rb_tree_node_base *__node, const _Rb_tree_node_base *__root) throw ()`
- `_Rb_tree_node_base * _Rb_tree_decrement (_Rb_tree_node_base *__x) throw ()`
- `const _Rb_tree_node_base * _Rb_tree_decrement (const _Rb_tree_node_base *__x) throw ()`
- `_Rb_tree_node_base * _Rb_tree_increment (_Rb_tree_node_base *__x) throw ()`
- `const _Rb_tree_node_base * _Rb_tree_increment (const _Rb_tree_node_base *__x) throw ()`
- `void _Rb_tree_insert_and_rebalance (const bool __insert_left, _Rb_tree_node_base *__x, _Rb_tree_node_base *__p, _Rb_tree_node_base &__header) throw ()`
- `_Rb_tree_node_base * _Rb_tree_rebalance_for_erase (_Rb_tree_node_base *const __z, _Rb_tree_node_base &__header) throw ()`
- `template<class _Dom >`
`_Expr< _UnClos< struct std:: _Abs, _Expr, _Dom >, typename _Dom::value_type > abs (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Tp abs (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std:: _Abs, _ValArray, _Tp >, _Tp > abs (const valarray< _Tp > &__v)`
- `constexpr double abs (double __x)`
- `constexpr float abs (float __x)`
- `long abs (long __i)`
- `constexpr long double abs (long double __x)`
- `long long abs (long long __x)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp accumulate (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp accumulate (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`

- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type acos (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Acos, _Expr, _Dom >, typename _Dom::value_type > acos (const _Expr< _↵`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`std::complex< _Tp > acos (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Acos, _ValArray, _Tp >, _Tp > acos (const valarray< _Tp > &__v)`
- `constexpr float acos (float __x)`
- `constexpr long double acos (long double __x)`
- `template<typename _Tp >`
`std::complex< _Tp > acosh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr _Tp * addressof (_Tp &__r) noexcept`
- `template<typename _Tp >`
`const _Tp * addressof (const _Tp &&)=delete`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __↵`
`result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __↵`
`result, _BinaryOperation __binary_op)`
- `template<typename _Filter >`
`constexpr _Filter adjacent_find (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`constexpr _Filter adjacent_find (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate ↵`
`__binary_pred)`
- `template<typename _InputIterator, typename _Distance >`
`constexpr void advance (_InputIterator &__i, _Distance __n)`
- `template<typename _CharT, typename _Distance >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, void >::__type advance (istreambuf_iterator< _↵`
`CharT > &__i, _Distance __n)`
- `void * align (size_t __align, size_t __size, void *&__ptr, size_t &__space) noexcept`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool all_of (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ValueType >`
`_ValueType any_cast (const any &__any)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool any_of (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Fn, typename _Tuple >`
`constexpr decltype(auto) apply (_Fn &&__f, _Tuple &&__t) noexcept(__unpack_std_tuple< is_nothrow_invocable,`
`_Fn, _Tuple >)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type arg (_Tp __x)`

- `template<typename _Tp >`
`_Tp arg (const complex< _Tp > &)`
- `template<typename _Tp, typename... _Up>`
`array (_Tp, _Up...) -> array< enable_if_t<(is_same_v< _Tp, _Up > &&...), _Tp >, 1+sizeof...(Up)>`
- `template<typename _Type, size_t _Extent>`
`span< const byte, _Extent==dynamic_extent ? dynamic_extent : _Extent *sizeof(_Type)> as_bytes (span< _Type, _Extent > __sp) noexcept`
- `template<typename _Tp >`
`constexpr add_const_t< _Tp > & as_const (_Tp &__t) noexcept`
- `template<typename _Tp >`
`void as_const (const _Tp &&)=delete`
- `template<typename _Type, size_t _Extent>`
`requires (is_const_v<_Type>)`
`span< byte, _Extent==dynamic_extent ? dynamic_extent : _Extent *sizeof(_Type)> as_writable_bytes (span< _Type, _Extent > __sp) noexcept`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type asin (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std:: _Asin, _Expr, _Dom >, typename _Dom::value_type > asin (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`std::complex< _Tp > asin (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std:: _Asin, _ValArray, _Tp >, _Tp > asin (const valarray< _Tp > &__v)`
- `constexpr float asin (float __x)`
- `constexpr long double asin (long double __x)`
- `template<typename _Tp >`
`std::complex< _Tp > asinh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<size_t _Align, class _Tp >`
`constexpr _Tp * assume_aligned (_Tp *__ptr) noexcept`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (_Fn &&__fn, _Args &&... __args)`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (launch __policy, _Fn &&__fn, _Args &&... __args)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type atan (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std:: _Atan, _Expr, _Dom >, typename _Dom::value_type > atan (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`std::complex< _Tp > atan (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std:: _Atan, _ValArray, _Tp >, _Tp > atan (const valarray< _Tp > &__v)`
- `constexpr float atan (float __x)`

- constexpr long double **atan** (long double __x)
- template<typename _Tp, typename _Up >
constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type **atan2** (_Tp __y, _Up __x)
- template<class _Dom >
_Expr< _BinClos< struct std:: Atan2, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type > **atan2** (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom::value_type &__t)
- template<class _Dom >
_Expr< _BinClos< struct std:: Atan2, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type > **atan2** (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)
- template<class _Dom1, class _Dom2 >
_Expr< _BinClos< struct std:: Atan2, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > **atan2** (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2::value_type > &__e2)
- template<class _Dom >
_Expr< _BinClos< struct std:: Atan2, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > **atan2** (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__e)
- template<typename _Tp >
_Expr< _BinClos< struct std:: Atan2, _Constant, _ValArray, _Tp, _Tp >, _Tp > **atan2** (const typename valarray< _Tp >::value_type &__t, const valarray< _Tp > &__v)
- template<typename _Tp >
_Expr< _BinClos< struct std:: Atan2, _ValArray, _Constant, _Tp, _Tp >, _Tp > **atan2** (const valarray< _Tp > &__v, const typename valarray< _Tp >::value_type &__t)
- template<typename _Tp >
_Expr< _BinClos< struct std:: Atan2, _ValArray, _ValArray, _Tp, _Tp >, _Tp > **atan2** (const valarray< _Tp > &__v, const valarray< _Tp > &__w)
- template<class _Dom >
_Expr< _BinClos< struct std:: Atan2, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > **atan2** (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)
- constexpr float **atan2** (float __y, float __x)
- constexpr long double **atan2** (long double __y, long double __x)
- template<typename _Tp >
std::complex< _Tp > **atanh** (const std::complex< _Tp > &__z)
- template<typename _ITp >
bool **atomic_compare_exchange_strong** (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept
- template<typename _ITp >
bool **atomic_compare_exchange_strong** (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept
- template<typename _ITp >
bool **atomic_compare_exchange_strong_explicit** (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
bool **atomic_compare_exchange_strong_explicit** (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
bool **atomic_compare_exchange_weak** (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept
- template<typename _ITp >
bool **atomic_compare_exchange_weak** (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept

- `template<typename _ITp >`
`bool atomic_compare_exchange_weak_explicit (atomic<_ITp> *__a, __atomic_val_t<_ITp> *__i1, __atomic_val_t<_ITp> __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_weak_explicit (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> *__i1, __atomic_val_t<_ITp> __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange_explicit (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange_explicit (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add_explicit (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add_explicit (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and_explicit (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and_explicit (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or_explicit (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or_explicit (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub_explicit (atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m) noexcept`

- `template<typename _ITp >`
`_ITp atomic_fetch_sub_explicit (volatile atomic<_ITp> *__a, __atomic_diff_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor_explicit (__atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor_explicit (volatile __atomic_base<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `void atomic_flag_clear (atomic_flag *__a) noexcept`
- `void atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `void atomic_flag_notify_all (atomic_flag *__a) noexcept`
- `void atomic_flag_notify_one (atomic_flag *__a) noexcept`
- `bool atomic_flag_test (const atomic_flag *__a) noexcept`
- `bool atomic_flag_test (const volatile atomic_flag *__a) noexcept`
- `bool atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool atomic_flag_test_explicit (const atomic_flag *__a, memory_order __m) noexcept`
- `bool atomic_flag_test_explicit (const volatile atomic_flag *__a, memory_order __m) noexcept`
- `void atomic_flag_wait (atomic_flag *__a, bool __old) noexcept`
- `void atomic_flag_wait_explicit (atomic_flag *__a, bool __old, memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_init (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`void atomic_init (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`bool atomic_is_lock_free (const atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`bool atomic_is_lock_free (const volatile atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load (const atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load (const volatile atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load_explicit (const atomic<_ITp> *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load_explicit (const volatile atomic<_ITp> *__a, memory_order __m) noexcept`
- `template<typename _Tp >`
`void atomic_notify_all (atomic<_Tp> *__a) noexcept`
- `template<typename _Tp >`
`void atomic_notify_one (atomic<_Tp> *__a) noexcept`
- `void atomic_signal_fence (memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_store (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`

- `template<typename _ITp >`
`void atomic_store (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`void atomic_store_explicit (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_store_explicit (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `void atomic_thread_fence (memory_order __m) noexcept`
- `template<typename _Tp >`
`void atomic_wait (const atomic< _Tp > *__a, typename std::atomic< _Tp >::value_type __old) noexcept`
- `template<typename _Tp >`
`void atomic_wait_explicit (const atomic< _Tp > *__a, typename std::atomic< _Tp >::value_type __old, std::memory_order __m) noexcept`
- `template<typename _Container >`
`constexpr back_insert_iterator< _Container > back_inserter (_Container &__x)`
- `template<typename _ForwardIterator >`
`basic_regex (_ForwardIterator, _ForwardIterator, regex_constants::syntax_option_type={}) -> basic_regex< typename iterator_traits< _ForwardIterator >::value_type >`
- `template<typename _InputIterator, typename _CharT = typename iterator_traits< _InputIterator >::value_type, typename _Allocator = allocator< _CharT >, typename = _RequireInputIter< _InputIterator >, typename = _RequireAllocator< _Allocator >>`
`basic_string (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> basic_string< _CharT, char_traits< _CharT >, _Allocator >`
- `template<typename _CharT, typename _Traits, typename _Allocator = allocator< _CharT >, typename = _RequireAllocator< _Allocator >>`
`basic_string (basic_string_view< _CharT, _Traits >, const _Allocator &=_Allocator()) -> basic_string< _CharT, _Traits, _Allocator >`
- `template<typename _CharT, typename _Traits, typename _Allocator = allocator< _CharT >, typename = _RequireAllocator< _Allocator >>`
`basic_string (basic_string_view< _CharT, _Traits >, typename basic_string< _CharT, _Traits, _Allocator >::size_type, typename basic_string< _CharT, _Traits, _Allocator >::size_type, const _Allocator &=_Allocator()) -> basic_string< _CharT, _Traits, _Allocator >`
- `template<contiguous_iterator _It, sized_sentinel_for< _It > _End>`
`basic_string_view (_It, _End) -> basic_string_view< iter_value_t< _It > >`
- `template<typename _Container >`
`constexpr auto begin (_Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * begin (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto begin (const _Container &__cont) -> decltype(__cont.begin())`
- `template<class _Tp >`
`const _Tp * begin (const valarray< _Tp > &__va) noexcept`
- `template<class _Tp >`
`_Tp * begin (valarray< _Tp > &__va) noexcept`
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type beta (_Tpa __a, _Tpb __b)`
- `float betaf (float __a, float __b)`
- `long double betal (long double __a, long double __b)`
- `template<typename _Filter, typename _Tp >`
`constexpr bool binary_search (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr bool binary_search (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr bool binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`

- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr bool binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _Func, typename... _BoundArgs>`
`constexpr _Bind_helper< __is_socketlike< _Func >::value, _Func, _BoundArgs... >::type bind (_Func &&__f, _BoundArgs &&... __args)`
- `template<typename _Result, typename _Func, typename... _BoundArgs>`
`constexpr _Bindres_helper< _Result, _Func, _BoundArgs... >::type bind (_Func &&__f, _BoundArgs &&... __args)`
- `template<typename _Operation, typename _Tp >`
`binder1st< _Operation > bind1st (const _Operation &__fn, const _Tp &__x)`
- `template<typename _Operation, typename _Tp >`
`binder2nd< _Operation > bind2nd (const _Operation &__fn, const _Tp &__x)`
- `template<typename _Fn, typename... _Args>`
`constexpr _Bind_front_t< _Fn, _Args... > bind_front (_Fn &&__fn, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Bind_front_t< _Fn, _Args... >, int, _Fn, _Args... >)`
- `template<typename _To, typename _From >`
`requires (sizeof(_To) == sizeof(_From)) && __is_trivially_copyable(_To) && __is_trivially_copyable(_From)`
`constexpr _To bit_cast (const _From &__from) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > bit_ceil (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > bit_floor (_Tp __x) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > bit_width (_Tp __x) noexcept`
- `ios_base & boolalpha (ios_base &__base)`
- `template<typename _Callable, typename... _Args>`
`void call_once (once_flag &__once, _Callable &&__f, _Args &&... __args)`
- `template<typename _Container >`
`constexpr auto cbegin (const _Container &__cont) noexcept(noexcept(std::begin(__cont))) -> decltype(std::begin(__cont))`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::value, double >::type ceil (_Tp __x)`
- `constexpr float ceil (float __x)`
- `constexpr long double ceil (long double __x)`
- `template<typename _Container >`
`constexpr auto cend (const _Container &__cont) noexcept(noexcept(std::end(__cont))) -> decltype(std::end(__cont))`
- `template<typename _Tp >`
`constexpr const _Tp & clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi, _Compare __comp)`
- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_greater (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_greater_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_less (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_less_equal (_Tp __t, _Up __u) noexcept`

- `template<typename _Tp, typename _Up >`
`constexpr bool cmp_not_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type comp_ellint_1 (_Tp __k)`
- `float comp_ellint_1f (float __k)`
- `long double comp_ellint_1l (long double __k)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type comp_ellint_2 (_Tp __k)`
- `float comp_ellint_2f (float __k)`
- `long double comp_ellint_2l (long double __k)`
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type comp_ellint_3 (_Tp __k, _Tpn __nu)`
- `float comp_ellint_3f (float __k, float __nu)`
- `long double comp_ellint_3l (long double __k, long double __nu)`
- `template<typename _Tp >`
`constexpr std::complex< typename __gnu_cxx::__promote< _Tp >::__type > conj (_Tp __x)`
- `template<typename _Tp >`
`constexpr complex< _Tp > conj (const complex< _Tp > &)`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp >`
`__shared_ptr< _Tp, _Lp > const_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, typename... _Args >`
`constexpr auto construct_at (_Tp * __location, _Args &&... __args) noexcept(noexcept(::new((void *) 0) <←
Tp(std::declval< _Args >()...))) -> decltype(::new((void *) 0) _Tp(std::declval< _Args >()...))`
- `template<typename _II, typename _OI >`
`constexpr _OI copy (_II __first, _II __last, _OI __result)`
- `template<typename _Iiter, typename _Oiter >`
`constexpr _Oiter copy (_Iiter, _Iiter, _Oiter)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type copy
(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, ostreambuf_iterator< _CharT
> __result)`
- `template<typename _BI1, typename _BI2 >`
`constexpr _BI2 copy_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _Biter1, typename _Biter2 >`
`constexpr _Biter2 copy_backward (_Biter1, _Biter1, _Biter2)`
- `template<typename _Iiter, typename _Oiter, typename _Predicate >`
`constexpr _Oiter copy_if (_Iiter, _Iiter, _Oiter, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate
__pred)`
- `template<typename _Iiter, typename _Size, typename _Oiter >`
`constexpr _Oiter copy_n (_Iiter, _Size, _Oiter)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`constexpr _OutputIterator copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type cos (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Cos, _Expr, _Dom >, typename _Dom::value_type > cos (const _Expr< _Dom,
typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > cos (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Cos, _ValArray, _Tp >, _Tp > cos (const valarray< _Tp > &__v)`

- constexpr float **cos** (float __x)
- constexpr long double **cos** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **cosh** (_Tp __x)
- template<class _Dom >
_Expr< _UnClos< struct std::_Cosh, _Expr, _Dom >, typename _Dom::value_type > **cosh** (const _Expr< _↵
_Dom, typename _Dom::value_type > &__e)
- template<typename _Tp >
complex< _Tp > **cosh** (const **complex**< _Tp > &)
- template<typename _Tp >
_Expr< _UnClos< struct std::_Cosh, _ValArray, _Tp >, _Tp > **cosh** (const **valarray**< _Tp > &__v)
- constexpr float **cosh** (float __x)
- constexpr long double **cosh** (long double __x)
- template<typename _Iter, typename _Tp >
constexpr **iterator_traits**< _Iter >::difference_type **count** (_Iter, _Iter, const _Tp &)
- template<typename _InputIterator, typename _Tp >
constexpr **iterator_traits**< _InputIterator >::difference_type **count** (_InputIterator __first, _InputIterator __last,
const _Tp &__value)
- template<typename _Iter, typename _Predicate >
constexpr **iterator_traits**< _Iter >::difference_type **count_if** (_Iter, _Iter, _Predicate)
- template<typename _InputIterator, typename _Predicate >
constexpr **iterator_traits**< _InputIterator >::difference_type **count_if** (_InputIterator __first, _InputIterator __last,
_Predicate __pred)
- template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > **countl_one** (_Tp __x) noexcept
- template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > **countl_zero** (_Tp __x) noexcept
- template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > **countr_one** (_Tp __x) noexcept
- template<typename _Tp >
constexpr _If_is_unsigned_integer< _Tp, int > **countr_zero** (_Tp __x) noexcept
- template<typename _Container >
constexpr auto **cbegin** (const _Container &__cont) -> decltype(std::rbegin(__cont))
- template<typename _Container >
constexpr auto **crend** (const _Container &__cont) -> decltype(std::rend(__cont))
- **exception_ptr** **current_exception** () noexcept
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_i** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_if** (float __nu, float __x)
- long double **cyl_bessel_il** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_j** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_jf** (float __nu, float __x)
- long double **cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_kf** (float __nu, float __x)
- long double **cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **cyl_neumannf** (float __nu, float __x)
- long double **cyl_neumannl** (long double __nu, long double __x)

- `template<typename _Container >`
`constexpr auto data (_Container &__cont) noexcept(noexcept(__cont.data())) -> decltype(__cont.data())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * data (_Tp(&__array)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto data (const _Container &__cont) noexcept(noexcept(__cont.data())) -> decltype(__cont.data())`
- `template<typename _Tp >`
`constexpr const _Tp * data (initializer_list< _Tp > __il) noexcept`
- `ios_base & dec (ios_base &__base)`
- `void declare_no_pointers (char *, size_t)`
- `void declare_reachable (void *)`
- `template<typename _Tp >`
`auto declval () noexcept -> decltype(__declval< _Tp > (0))`
- `ios_base & defaultfloat (ios_base &__base)`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`deque (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> deque< _ValT, _Allocator >`
- `template<typename _ForwardIterator >`
`constexpr void destroy (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _Tp >`
`constexpr void destroy_at (_Tp *__location)`
- `template<typename _ForwardIterator, typename _Size >`
`constexpr _ForwardIterator destroy_n (_ForwardIterator __first, _Size __count)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::difference_type distance (_InputIterator __first, _InputIterator __last)`
- `ldiv_t div (long __i, long __j)`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > dynamic_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type ellint_1 (_Tp __k, _Tpp __phi)`
- `float ellint_1f (float __k, float __phi)`
- `long double ellint_1l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type ellint_2 (_Tp __k, _Tpp __phi)`
- `float ellint_2f (float __k, float __phi)`
- `long double ellint_2l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpn, typename _Tpp >`
`__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi)`
- `float ellint_3f (float __k, float __nu, float __phi)`
- `long double ellint_3l (long double __k, long double __nu, long double __phi)`
- `template<typename _Container >`
`constexpr auto empty (const _Container &__cont) noexcept(noexcept(__cont.empty())) -> decltype(__cont.empty())`
- `template<typename _Tp, size_t _Nm>`
`constexpr bool empty (const _Tp(&)[_Nm]) noexcept`
- `template<typename _Tp >`
`constexpr bool empty (initializer_list< _Tp > __il) noexcept`
- `template<typename _Container >`
`constexpr auto end (_Container &__cont) -> decltype(__cont.end())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * end (_Tp(&__arr)[_Nm]) noexcept`

- `template<typename _Container >`
`constexpr auto end (const _Container &__cont) -> decltype(__cont.end())`
- `template<class _Tp >`
`const _Tp * end (const valarray< _Tp > &__va) noexcept`
- `template<class _Tp >`
`_Tp * end (valarray< _Tp > &__va) noexcept`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & endl (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & ends (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _II1, typename _II2 >`
`constexpr bool equal (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _II1, typename _II2 >`
`constexpr bool equal (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >`
`constexpr bool equal (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >`
`constexpr bool equal (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _IIter2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _IIter1, typename _IIter2 >`
`constexpr bool equal (_IIter1, _IIter1, _IIter2)`
- `template<typename _Filter, typename _Tp >`
`constexpr pair< _Filter, _Filter > equal_range (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr pair< _Filter, _Filter > equal_range (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr pair< _ForwardIterator, _ForwardIterator > equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Up >`
`constexpr basic_string< _CharT, _Traits, _Alloc >::size_type erase (basic_string< _CharT, _Traits, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`deque< _Tp, _Alloc >::size_type erase (deque< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`forward_list< _Tp, _Alloc >::size_type erase (forward_list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`list< _Tp, _Alloc >::size_type erase (list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`constexpr vector< _Tp, _Alloc >::size_type erase (vector< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Predicate >`
`constexpr basic_string< _CharT, _Traits, _Alloc >::size_type erase_if (basic_string< _CharT, _Traits, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`deque< _Tp, _Alloc >::size_type erase_if (deque< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`forward_list< _Tp, _Alloc >::size_type erase_if (forward_list< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`list< _Tp, _Alloc >::size_type erase_if (list< _Tp, _Alloc > &__cont, _Predicate __pred)`

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`map< _Key, _Tp, _Compare, _Alloc >::size_type erase_if (map< _Key, _Tp, _Compare, _Alloc > &__cont,`
`_Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`multimap< _Key, _Tp, _Compare, _Alloc >::size_type erase_if (multimap< _Key, _Tp, _Compare, _Alloc >`
`&__cont, _Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`multiset< _Key, _Compare, _Alloc >::size_type erase_if (multiset< _Key, _Compare, _Alloc > &__cont, _↵`
`Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`set< _Key, _Compare, _Alloc >::size_type erase_if (set< _Key, _Compare, _Alloc > &__cont, _Predicate _↵`
`pred)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`unordered_map< _Key, _Tp, _Hash, _CPred, _Alloc >::size_type erase_if (unordered_map< _Key, _Tp, _Hash,`
`_CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`unordered_multimap< _Key, _Tp, _Hash, _CPred, _Alloc >::size_type erase_if (unordered_multimap< _Key,`
`_Tp, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`unordered_multiset< _Key, _Hash, _CPred, _Alloc >::size_type erase_if (unordered_multiset< _Key, _Hash,`
`_CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`unordered_set< _Key, _Hash, _CPred, _Alloc >::size_type erase_if (unordered_set< _Key, _Hash, _CPred,`
`_Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`constexpr vector< _Tp, _Alloc >::size_type erase_if (vector< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Up = _Tp>`
`constexpr _Tp exchange (_Tp &__obj, _Up &&__new_val) noexcept(__and< is_nothrow_move_constructible<`
`_Tp >, is_nothrow_assignable< _Tp &, _Up > >::value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result,`
`_Tp __init)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _OutputIterator exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result,`
`_Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type exp (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Exp, _Expr, _Dom >, typename _Dom::value_type > exp (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > exp (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Exp, _ValArray, _Tp >, _Tp > exp (const valarray< _Tp > &__v)`
- `constexpr float exp (float __x)`
- `constexpr long double exp (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type expint (_Tp __x)`
- `float expintf (float __x)`
- `long double expintl (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type fabs (_Tp __x)`

- `template<typename _Tp >`
`_Tp fabs (const std::complex< _Tp > &__z)`
- `constexpr float fabs (float __x)`
- `constexpr long double fabs (long double __x)`
- `template<typename _Filter, typename _Tp >`
`constexpr void fill (_Filter, _Filter, const _Tp &)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _OI, typename _Size, typename _Tp >`
`constexpr _OI fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _OIter, typename _Size, typename _Tp >`
`constexpr _OIter fill_n (_OIter, _Size, const _Tp &)`
- `template<typename _Iter, typename _Tp >`
`constexpr _Iter find (_Iter, _Iter, const _Tp &)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _InputIterator find (_InputIterator __first, _InputIterator __last, const _Tp &__val)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT > >::__type find`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT &__val)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 find_end (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 find_end (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2`
`__first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2`
`__first2, _ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 find_first_of (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 find_first_of (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`constexpr _InputIterator find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, ↵`
`_ForwardIterator __last2)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _InputIterator find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, ↵`
`_ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _Iter, typename _Predicate >`
`constexpr _Iter find_if (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Iter, typename _Predicate >`
`constexpr _Iter find_if_not (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `ios_base & fixed (ios_base &__base)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type floor (_Tp __x)`
- `constexpr float floor (float __x)`
- `constexpr long double floor (long double __x)`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & flush (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type fmod (_Tp __x, _Up __y)`
- `constexpr float fmod (float __x, float __y)`
- `constexpr long double fmod (long double __x, long double __y)`
- `template<typename _Iter, typename _Funct >`
`constexpr _Funct for_each (_Iter, _Iter, _Funct)`
- `template<typename _InputIterator, typename _Function >`
`constexpr _Function for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _InputIterator, typename _Size, typename _Function >`
`constexpr _InputIterator for_each_n (_InputIterator __first, _Size __n, _Function __f)`
- `template<typename _Tp >`
`constexpr _Tp && forward (typename std::remove_reference< _Tp >::__type &&__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && forward (typename std::remove_reference< _Tp >::__type &__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > forward_as_tuple (_Elements &&... __args) noexcept`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`forward_list (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> forward_list< _ValT, _Allocator >`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type frexp (_Tp __x, int *__exp)`
- `float frexp (float __x, int *__exp)`
- `long double frexp (long double __x, int *__exp)`
- `template<typename _Tp >`
`__detail::__integer_from_chars_result_type< _Tp > from_chars (const char * __first, const char * __last, _Tp & __value, int __base=10)`
- `template<typename _Container >`
`constexpr front_insert_iterator< _Container > front_inserter (_Container &__x)`
- `template<typename _Function, typename _Signature = typename __function_guide_helper<decltype(&_Function::operator())>::__type>`
`function (_Function) -> function< _Signature >`
- `template<typename _Res, typename... _ArgTypes>`
`function (_Res(*)(_ArgTypes...)) -> function< _Res(_ArgTypes...)>`
- `const error_category & future_category () noexcept`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > gcd (_Mn __m, _Nn __n) noexcept`
- `template<typename _Filter, typename _Generator >`
`constexpr void generate (_Filter, _Filter, _Generator)`
- `template<typename _ForwardIterator, typename _Generator >`
`constexpr void generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >`
`_RealType generate_canonical (_UniformRandomNumberGenerator &__g)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`constexpr _OIter generate_n (_OIter, _Size, _Generator)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`constexpr _OutputIterator generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`
- `const error_category & generic_category () noexcept`
- `template<std::size_t __Int, typename _Tp, std::size_t __Nm>`
`constexpr _Tp && get (array< _Tp, __Nm > &&__arr) noexcept`
- `template<std::size_t __Int, typename _Tp, std::size_t __Nm>`
`constexpr _Tp & get (array< _Tp, __Nm > &__arr) noexcept`

- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp && get (const array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp & get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > && get (const tuple< _Elements... > &&__t)`
`noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > & get (const tuple< _Elements... > &__t)`
`noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp && get (const tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & get (const tuple< _Types... > &__t) noexcept`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > const && get (const variant< _Types... > &&)`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp && get (const variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr const variant_alternative_t< _Np, variant< _Types... > > && get (const variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > const & get (const variant< _Types... > &)`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & get (const variant< _Types... > &__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr const variant_alternative_t< _Np, variant< _Types... > > & get (const variant< _Types... > &__v)`
- `template<size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > && get (tuple< _Elements... > &&__t) noexcept`
- `template<size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > & get (tuple< _Elements... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & get (tuple< _Types... > &__t) noexcept`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > && get (variant< _Types... > &&)`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && get (variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > & get (variant< _Types... > &)`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & get (variant< _Types... > &__v)`
- **Catalogs & `get_catalogs` ()**
- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<size_t _Np, typename... _Types>`
`constexpr add_pointer_t< const variant_alternative_t< _Np, variant< _Types... > > > > get_if (const variant< _Types... > *__ptr) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr add_pointer_t< const _Tp > get_if (const variant< _Types... > *__ptr) noexcept`
- `template<size_t _Np, typename... _Types>`
`constexpr add_pointer_t< variant_alternative_t< _Np, variant< _Types... > > > > get_if (variant< _Types... > *__ptr) noexcept`

- `template<typename _Tp, typename... _Types>`
`constexpr add_pointer_t< _Tp > get_if (variant< _Types... > *__ptr) noexcept`
- `template<typename _MoneyT >`
`_Get_money< _MoneyT > get_money (_MoneyT &__mon, bool __intl=false)`
- `new_handler get_new_handler () noexcept`
- `pointer_safety get_pointer_safety () noexcept`
- `template<typename _Tp >`
`pair< _Tp *, ptrdiff_t > get_temporary_buffer (ptrdiff_t __len) noexcept`
- `terminate_handler get_terminate () noexcept`
- `template<typename _CharT >`
`_Get_time< _CharT > get_time (std::tm *__tmb, const _CharT *__fmt)`
- `unexpected_handler get_unexpected () noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str, _CharT __delim)`
- `template<> basic_istream< char > & getline (basic_istream< char > &&__in, basic_string< char > &__str, char __delim)`
- `template<> basic_istream< wchar_t > & getline (basic_istream< wchar_t > &&__in, basic_string< wchar_t > &__str, wchar_t __delim)`
- `template<typename _Facet >`
`bool has_facet (const locale &__loc) throw ()`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, bool > has_single_bit (_Tp __x) noexcept`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type hermite (unsigned int __n, _Tp __x)`
- `float hermitef (unsigned int __n, float __x)`
- `long double hermite (unsigned int __n, long double __x)`
- `ios_base & hex (ios_base &__base)`
- `ios_base & hexfloat (ios_base &__base)`
- `template<typename _Tp, typename... _Types>`
`constexpr bool holds_alternative (const variant< _Types... > &__v) noexcept`
- `template<typename _Tp, typename _Up, typename _Vp >`
`__gnu_cxx::__promoted_t< _Tp, _Up, _Vp > hypot (_Tp __x, _Up __y, _Vp __z)`
- `double hypot (double __x, double __y, double __z)`
- `float hypot (float __x, float __y, float __z)`
- `long double hypot (long double __x, long double __y, long double __z)`

- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type imag (_Tp)`
- `template<typename _Tp >`
`constexpr _Tp imag (const complex< _Tp > &__z)`
- `template<typename _Up, typename _Tp >`
`constexpr bool in_range (_Tp __t) noexcept`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool includes (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`constexpr bool includes (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr bool includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`constexpr bool includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _Tp >`
`constexpr _OutputIterator inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2 >`
`constexpr _Tp inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`
- `template<typename _BidirectionalIterator >`
`void inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Biter >`
`void inplace_merge (_Biter, _Biter, _Biter)`
- `template<typename _Biter, typename _Compare >`
`void inplace_merge (_Biter, _Biter, _Biter, _Compare)`
- `template<typename _Container >`
`constexpr insert_iterator< _Container > inserter (_Container &__x, std::__detail::__range_iter_t< _Container > __i)`
- `ios_base & internal (ios_base &__base)`
- `template<typename _Callable, typename... _Args>`
`constexpr invoke_result_t< _Callable, _Args... > invoke (_Callable &&__fn, _Args &&... __args) noexcept(is_nothrow_invocable_v< _Callable, _Args... >)`
- `const error_category & iostream_category () noexcept`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void iota (_ForwardIterator __first, _ForwardIterator __last, _Tp __value)`
- `constexpr bool is_constant_evaluated () noexcept`
- `template<typename _S1, typename _S2, typename _M1, typename _M2 >`
`constexpr bool is_corresponding_member (_M1 _S1::*__m1, _M2 _S2::*__m2) noexcept`
- `constexpr bool is_eq (partial_ordering __cmp) noexcept`

- constexpr bool **is_gt** (partial_ordering __cmp) noexcept
- constexpr bool **is_gteq** (partial_ordering __cmp) noexcept
- template<typename _RAIter >
constexpr bool **is_heap** (_RAIter, _RAIter)
- template<typename _RAIter, typename _Compare >
constexpr bool **is_heap** (_RAIter, _RAIter, _Compare)
- template<typename _RandomAccessIterator >
constexpr bool **is_heap** (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _RandomAccessIterator, typename _Compare >
constexpr bool **is_heap** (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- template<typename _RAIter >
constexpr _RAIter **is_heap_until** (_RAIter, _RAIter)
- template<typename _RAIter, typename _Compare >
constexpr _RAIter **is_heap_until** (_RAIter, _RAIter, _Compare)
- template<typename _RandomAccessIterator >
constexpr _RandomAccessIterator **is_heap_until** (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _RandomAccessIterator, typename _Compare >
constexpr _RandomAccessIterator **is_heap_until** (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- constexpr bool **is_lt** (partial_ordering __cmp) noexcept
- constexpr bool **is_lteq** (partial_ordering __cmp) noexcept
- constexpr bool **is_neq** (partial_ordering __cmp) noexcept
- template<typename _Iter, typename _Predicate >
constexpr bool **is_partitioned** (_Iter, _Iter, _Predicate)
- template<typename _InputIterator, typename _Predicate >
constexpr bool **is_partitioned** (_InputIterator __first, _InputIterator __last, _Predicate __pred)
- template<typename _Filter1, typename _Filter2 >
constexpr bool **is_permutation** (_Filter1, _Filter1, _Filter2)
- template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >
constexpr bool **is_permutation** (_Filter1, _Filter1, _Filter2, _BinaryPredicate)
- template<typename _ForwardIterator1, typename _ForwardIterator2 >
constexpr bool **is_permutation** (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)
- template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >
constexpr bool **is_permutation** (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _BinaryPredicate __pred)
- template<typename _ForwardIterator1, typename _ForwardIterator2 >
constexpr bool **is_permutation** (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2)
- template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >
constexpr bool **is_permutation** (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __pred)
- template<typename _Tp, typename _Mem >
constexpr bool **is_pointer_interconvertible_with_class** (_Mem _Tp::*__mp) noexcept
- template<typename _Filter >
constexpr bool **is_sorted** (_Filter, _Filter)
- template<typename _Filter, typename _Compare >
constexpr bool **is_sorted** (_Filter, _Filter, _Compare)
- template<typename _ForwardIterator >
constexpr bool **is_sorted** (_ForwardIterator __first, _ForwardIterator __last)
- template<typename _ForwardIterator, typename _Compare >
constexpr bool **is_sorted** (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)

- `template<typename _Filter >`
`constexpr _Filter is_sorted_until (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr _Filter is_sorted_until (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _CharT >`
`bool isalnum (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isalpha (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isblank (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool iscntrl (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isgraph (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool islower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isprint (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool ispunct (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isspace (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isupper (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isxdigit (_CharT __c, const locale &__loc)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr void iter_swap (_Filter1, _Filter2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr void iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`
- `template<typename _Tp >`
`_Tp kill_dependency (_Tp __y) noexcept`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type laguerre (unsigned int __n, _Tp __x)`
- `float laguerref (unsigned int __n, float __x)`
- `long double laguerrel (unsigned int __n, long double __x)`
- `template<typename _Ret, typename... _Args>`
`void launder (_Ret*)(_Args...)=delete`
- `template<typename _Ret, typename... _Args>`
`void launder (_Ret*)(_Args.....)=delete`
- `template<typename _Tp >`
`constexpr _Tp * launder (_Tp *__p) noexcept`
- `void launder (const void *)=delete`
- `void launder (const volatile void *)=delete`
- `void launder (void *)=delete`
- `void launder (volatile void *)=delete`

- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > lcm (_Mn __m, _Nn __n) noexcept`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type ldexp (_Tp __x, int __exp)`
- `constexpr float ldexp (float __x, int __exp)`
- `constexpr long double ldexp (long double __x, int __exp)`
- `ios_base & left (ios_base & __base)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type legendre (unsigned int __l, _Tp __x)`
- `float legendref (unsigned int __l, float __x)`
- `long double legendrel (unsigned int __l, long double __x)`
- `template<typename _Tp, typename _Up, typename _Vp >`
`constexpr __gnu_cxx::__promote_t< _Tp, _Up, _Vp > lerp (_Tp __x, _Up __y, _Vp __z) noexcept`
- `constexpr double lerp (double __a, double __b, double __t) noexcept`
- `constexpr float lerp (float __a, float __b, float __t) noexcept`
- `constexpr long double lerp (long double __a, long double __b, long double __t) noexcept`
- `template<typename _I1, typename _I2 >`
`constexpr bool lexicographical_compare (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _I2, typename _Compare >`
`constexpr bool lexicographical_compare (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, _Compare __comp)`
- `template<typename _Iiter1, typename _Iiter2 >`
`constexpr bool lexicographical_compare (_Iiter1, _Iiter1, _Iiter2, _Iiter2)`
- `template<typename _Iiter1, typename _Iiter2, typename _Compare >`
`constexpr bool lexicographical_compare (_Iiter1, _Iiter1, _Iiter2, _Iiter2, _Compare)`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`list (_InputIterator, _InputIterator, _Allocator= _Allocator()) -> list< _ValT, _Allocator >`
- `template<typename _L1, typename _L2, typename... _L3>`
`void lock (_L1 & __l1, _L2 & __l2, _L3 &... __l3)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type log (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::Log, _Expr, _Dom >, typename _Dom::value_type > log (const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<typename _Tp >`
`complex< _Tp > log (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::Log, _ValArray, _Tp >, _Tp > log (const valarray< _Tp > & __v)`
- `constexpr float log (float __x)`
- `constexpr long double log (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type log10 (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::Log10, _Expr, _Dom >, typename _Dom::value_type > log10 (const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<typename _Tp >`
`complex< _Tp > log10 (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::Log10, _ValArray, _Tp >, _Tp > log10 (const valarray< _Tp > & __v)`
- `constexpr float log10 (float __x)`
- `constexpr long double log10 (long double __x)`

- `template<typename _Filter, typename _Tp >`
`constexpr _Filter lower_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr _Filter lower_bound (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, ←
_Compare __comp)`
- `template<typename _Tp, typename... _Args>`
`enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, _Args... >, any > make_any (_Args &&... __args)`
- `template<typename _Tp, typename _Up, typename... _Args>`
`enable_if_t< is_constructible_v< any, in_place_type_t< _Tp >, initializer_list< _Up > &, _Args... >, any > make_any (initializer_list< _Up > __il, _Args &&... __args)`
- `error_code make_error_code (future_errc __errc) noexcept`
- `error_code make_error_code (io_errc __e) noexcept`
- `error_condition make_error_condition (future_errc __errc) noexcept`
- `error_condition make_error_condition (io_errc __e) noexcept`
- `template<typename _Ex >`
`exception_ptr make_exception_ptr (_Ex __ex) noexcept`
- `template<typename _Tp, typename _Tuple >`
`constexpr _Tp make_from_tuple (_Tuple && __t) noexcept(__unpack_std_tuple< is_nothrow_constructible, _Tp, _Tuple >)`
- `template<typename _RAIter >`
`constexpr void make_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void make_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`constexpr void make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Iterator >`
`constexpr move_iterator< _Iterator > make_move_iterator (_Iterator __i)`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp make_obj_using_allocator (const _Alloc & __a, _Args &&... __args)`
- `template<typename _Tp, typename... _Args>`
`constexpr enable_if_t< is_constructible_v< _Tp, _Args... >, optional< _Tp > > make_optional (_Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, _Args... >)`
- `template<typename _Tp >`
`constexpr enable_if_t< is_constructible_v< decay_t< _Tp >, _Tp >, optional< decay_t< _Tp > > > make_←
_optional (_Tp && __t) noexcept(is_nothrow_constructible_v< optional< decay_t< _Tp > >, _Tp >)`
- `template<typename _Tp, typename _Up, typename... _Args>`
`constexpr enable_if_t< is_constructible_v< _Tp, initializer_list< _Up > &, _Args... >, optional< _Tp > > make_optional (initializer_list< _Up > __il, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, initializer_list< _Up > &, _Args... >)`
- `template<typename _Iterator >`
`constexpr reverse_iterator< _Iterator > make_reverse_iterator (_Iterator __i)`
- `template<typename... _Elements>`
`constexpr tuple< typename __decay_and_strip< _Elements >::__type... > make_tuple (_Elements &&... __args)`

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`map (_InputIterator, _InputIterator, _Allocator) -> map< __iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, less< __iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less< __iter_key_t<_InputIterator>>, typename _Allocator = allocator< __iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`map (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> map< __iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Compare, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`map (initializer_list< pair< _Key, _Tp>>, _Allocator) -> map< _Key, _Tp, less< _Key>, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Compare = less< _Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`map (initializer_list< pair< _Key, _Tp>>, _Compare=_Compare(), _Allocator=_Allocator()) -> map< _Key, _Tp, _Compare, _Allocator>`
- `template<typename _Tp>`
`constexpr const _Tp & max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare>`
`constexpr const _Tp & max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp>`
`constexpr _Tp max (initializer_list< _Tp>)`
- `template<typename _Tp, typename _Compare>`
`constexpr _Tp max (initializer_list< _Tp>, _Compare)`
- `template<typename _Filter>`
`constexpr _Filter max_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare>`
`constexpr _Filter max_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator>`
`constexpr _ForwardIterator max_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare>`
`constexpr _ForwardIterator max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp, typename _Class>`
`constexpr _Mem_fn< _Tp _Class::*> mem_fn (_Tp _Class::* __pm) noexcept`
- `template<typename _Ret, typename _Tp>`
`const_mem_fun_t< _Ret, _Tp> mem_fun (_Ret(_Tp::* __f)() const)`
- `template<typename _Ret, typename _Tp>`
`mem_fun_t< _Ret, _Tp> mem_fun (_Ret(_Tp::* __f)())`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`const_mem_fun1_t< _Ret, _Tp, _Arg> mem_fun (_Ret(_Tp::* __f)(_Arg) const)`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_t< _Ret, _Tp, _Arg> mem_fun (_Ret(_Tp::* __f)(_Arg))`
- `template<typename _Ret, typename _Tp>`
`const_mem_fun_ref_t< _Ret, _Tp> mem_fun_ref (_Ret(_Tp::* __f)() const)`
- `template<typename _Ret, typename _Tp>`
`mem_fun_ref_t< _Ret, _Tp> mem_fun_ref (_Ret(_Tp::* __f)())`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`const_mem_fun1_ref_t< _Ret, _Tp, _Arg> mem_fun_ref (_Ret(_Tp::* __f)(_Arg) const)`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_ref_t< _Ret, _Tp, _Arg> mem_fun_ref (_Ret(_Tp::* __f)(_Arg))`
- `void * memchr (void *__s, int __c, size_t __n)`
- `template<typename _Iiter1, typename _Iiter2, typename _Oiter>`
`constexpr _Oiter merge (_Iiter1, _Iiter1, _Iiter2, _Iiter2, _Oiter)`

- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _↵`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _↵`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Tp >`
`constexpr enable_if_t< is_object_v< _Tp >, _Tp * > midpoint (_Tp *__a, _Tp *__b) noexcept`
- `template<typename _Tp >`
`constexpr enable_if_t< __and_v< is_arithmetic< _Tp >, is_same< remove_cv_t< _Tp >, _Tp >, __not<`
`is_same< _Tp, bool > >, _Tp > midpoint (_Tp __a, _Tp __b) noexcept`
- `template<typename _Tp >`
`constexpr const _Tp & min (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & min (const _Tp & __a, const _Tp & __b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr _Tp min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr _Tp min (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`constexpr _Filter min_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr _Filter min_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator min_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`constexpr pair< const _Tp &, const _Tp & > minmax (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< const _Tp &, const _Tp & > minmax (const _Tp & __a, const _Tp & __b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr pair< _Tp, _Tp > minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< _Tp, _Tp > minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`constexpr pair< _Filter, _Filter > minmax_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr pair< _Filter, _Filter > minmax_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr pair< _ForwardIterator, _ForwardIterator > minmax_element (_ForwardIterator __first, _Forward↵`
`Iterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > minmax_element (_ForwardIterator __first, _Forward↵`
`Iterator __last, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr pair< _Iter1, _Iter2 > mismatch (_Iter1, _Iter1, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`constexpr pair< _Iter1, _Iter2 > mismatch (_Iter1, _Iter1, _Iter2, _BinaryPredicate)`

- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `float modf (float __x, float * __iptr)`
- `long double modf (long double __x, long double * __iptr)`
- `template<typename _II, typename _OI >`
`constexpr _OI move (_II __first, _II __last, _OI __result)`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && move (_Tp && __t) noexcept`
- `template<typename _BI1, typename _BI2 >`
`constexpr _BI2 move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _Tp >`
`constexpr __conditional_t< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && > move_if_noexcept`
`(_Tp & __x) noexcept`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator< _Allocator>>`
`multimap (_InputIterator, _InputIterator, _Allocator) -> multimap< __iter_key_t< _InputIterator >, __iter_val_t<`
`_InputIterator >, less< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less< __iter_key_t<_InputIterator>>, typename _Allocator = allocator<↵`
`__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, type-`
`name = _RequireAllocator< _Allocator>>`
`multimap (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap< __iter↵`
`__key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Compare, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator< _Allocator>>`
`multimap (initializer_list< pair< _Key, _Tp > >, _Allocator) -> multimap< _Key, _Tp, less< _Key >, _Allocator`
`>`
- `template<typename _Key, typename _Tp, typename _Compare = less< _Key >, typename _Allocator = allocator<pair<const _Key, ↵`
`_Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator< _Allocator>>`
`multimap (initializer_list< pair< _Key, _Tp > >, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap<`
`_Key, _Tp, _Compare, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator< _Allocator>>`
`multiset (_InputIterator, _InputIterator, _Allocator) -> multiset< typename iterator_traits< _InputIterator >↵`
`::value_type, less< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename ↵`
`_Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocator<_Compare>, typename = _RequireAllocator< _Allocator>>`
`multiset (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> multiset< typename`
`iterator_traits< _InputIterator >::value_type, _Compare, _Allocator >`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator< _Allocator>>`
`multiset (initializer_list< _Key >, _Allocator) -> multiset< _Key, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Compare = less< _Key >, typename _Allocator = allocator<_Key>, typename = _RequireNot↵`
`_Allocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`multiset (initializer_list< _Key >, _Compare=_Compare(), _Allocator=_Allocator()) -> multiset< _Key, ↵`
`_Compare, _Allocator >`

- `template<typename _InputIterator >`
`constexpr _InputIterator next (_InputIterator __x, typename iterator_traits< _InputIterator >::difference_type __n=1)`
- `template<typename _BidirectionalIterator >`
`constexpr bool next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Biter >`
`constexpr bool next_permutation (_Biter, _Biter)`
- `template<typename _Biter, typename _Compare >`
`constexpr bool next_permutation (_Biter, _Biter, _Compare)`
- `ios_base & noboolalpha (ios_base & __base)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool none_of (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type norm (_Tp __x)`
- `template<typename _Tp >`
`_Tp constexpr norm (const complex< _Tp > &)`
- `template<typename _Tp >`
`constexpr _Tp norm (const complex< _Tp > & __z)`
- `ios_base & noshowbase (ios_base & __base)`
- `ios_base & noshowpoint (ios_base & __base)`
- `ios_base & noshowpos (ios_base & __base)`
- `ios_base & noskipws (ios_base & __base)`
- `template<typename _Predicate >`
`constexpr unary_negate< _Predicate > not1 (const _Predicate & __pred)`
- `template<typename _Predicate >`
`constexpr binary_negate< _Predicate > not2 (const _Predicate & __pred)`
- `template<typename _Fn >`
`constexpr auto not_fn (_Fn && __fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn && >::value)`
- `void notify_all_at_thread_exit (condition_variable &, unique_lock< mutex >)`
- `ios_base & nounitbuf (ios_base & __base)`
- `ios_base & nouppercase (ios_base & __base)`
- `template<typename _RAIter >`
`constexpr void nth_element (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void nth_element (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`constexpr void nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Compare __comp)`
- `ios_base & oct (ios_base & __base)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`

- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator!= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator!= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__not_equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__not_equal_to, typename _Dom::value_type >::result_type > operator!= (const _<`
`Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__not_equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__not_equal_to, typename _Dom::value_type >::result_type > operator!= (const _<`
`Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__not_equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std:<`
`::__not_equal_to, typename _Dom1::value_type >::result_type > operator!= (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator!= (const _Up & __lhs, const optional< _Tp > &__rhs) -> __optional_ne_t< _Up, _Tp >`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator!= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const basic_string< _CharT, _Traits, _<`
`_Alloc > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _StateT >`
`bool operator!= (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator!= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc >`
`&__y)`
- `template<typename _Bi_iter, class _Alloc >`
`bool operator!= (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__<`
`__m2)`
- `template<typename _Iterator >`
`constexpr bool operator!= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr bool operator!= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator!= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare,`
`_Alloc > &__y)`

- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator!= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc >`
`&__y)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator!= (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_ne_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator!= (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_ne_t<`
`_Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool operator!= (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator!= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() != __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator!= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__`
`__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator!= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool operator!= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `bool operator!= (const std::bernoulli_distribution &__d1, const std::bernoulli_distribution &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::binomial_distribution< _IntType > &__d1, const std::binomial_distribution< _IntType`
`> &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::cauchy_distribution< _RealType > &__d1, const std::cauchy_distribution< _RealType`
`> &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::chi_squared_distribution< _RealType > &__d1, const std::chi_squared_distribution<`
`_RealType > &__d2)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r>`
`bool operator!= (const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__lhs, const`
`std::discard_block_engine< _RandomNumberEngine, __p, __r > &__rhs)`
- `template<typename _IntType >`
`bool operator!= (const std::discrete_distribution< _IntType > &__d1, const std::discrete_distribution< _IntType`
`> &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution<`
`_RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution<`
`_RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::fisher_f_distribution< _RealType > &__d1, const std::fisher_f_distribution< _Real`
`Type > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::gamma_distribution< _RealType > &__d1, const std::gamma_distribution< _Real`
`Type > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _Int`
`Type > &__d2)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType >`
`bool operator!= (const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__lhs, const`
`std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__rhs)`

- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>`
`bool operator!= (const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__lhs, const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__rhs)`
- `template<typename _RealType >`
`bool operator!= (const std::lognormal_distribution< _RealType > &__d1, const std::lognormal_distribution< _RealType > &__d2)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>`
`bool operator!= (const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__lhs, const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__rhs)`
- `template<typename _IntType >`
`bool operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2)`
- `template<typename _RandomNumberEngine, size_t __k>`
`bool operator!= (const std::shuffle_order_engine< _RandomNumberEngine, __k > &__lhs, const std::shuffle_order_engine< _RandomNumberEngine, __k > &__rhs)`
- `template<typename _RealType >`
`bool operator!= (const std::student_t_distribution< _RealType > &__d1, const std::student_t_distribution< _RealType > &__d2)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r>`
`bool operator!= (const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__lhs, const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__rhs)`
- `template<typename _IntType >`
`bool operator!= (const std::uniform_int_distribution< _IntType > &__d1, const std::uniform_int_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::uniform_real_distribution< _IntType > &__d1, const std::uniform_real_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _RealType > &__d2)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator!= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_not_equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_not_equal_to, typename _Dom::value_type >::result_type > operator!= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator!= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`

- `template<typename _Tp, typename _Dp >`
`bool operator!= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__not_equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__not_equal_to, typename _Dom::value_type >::result_type > operator!= (const`
`valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename... _Types>`
`constexpr bool operator!= (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator!= (monostate, monostate) noexcept`
- `template<typename _Tp >`
`constexpr bool operator!= (nullptr_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator!= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool operator!= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__modulus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__modulus, typename _Dom::value_type >::result_type > operator% (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__modulus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__modulus, typename _Dom::value_type >::result_type > operator% (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__modulus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__modulus, typename _Dom1::value_type >::result_type > operator% (const _Expr< _Dom1, typename _`
`Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__modulus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__modulus, typename _Dom::value_type >::result_type > operator% (const typename _`
`Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__modulus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__modulus, typename _Dom::value_type >::result_type > operator% (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr _ios_Fmtflags operator& (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_ostate operator& (_ios_ostate __a, _ios_ostate __b)`
- `constexpr _ios_Openmode operator& (_ios_Openmode __a, _ios_Openmode __b)`
- `constexpr byte operator& (byte __l, byte __r) noexcept`
- `constexpr chars_format operator& (chars_format __lhs, chars_format __rhs) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > operator& (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > operator& (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`::__bitwise_and, typename _Dom1::value_type >::result_type > operator& (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > operator& (const type-`
`name _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > operator& (const`
`valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr launch operator& (launch __x, launch __y) noexcept`
- `constexpr memory_order operator& (memory_order __m, __memory_order_modifier __mod)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__logical_and, typename _Dom::value_type >::result_type > operator&& (const _Expr< __`
`Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__logical_and, typename _Dom::value_type >::result_type > operator&& (const __`
`Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`logical_and, typename _Dom1::value_type >::result_type > operator&& (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__logical_and, typename _Dom::value_type >::result_type > operator&& (const type-`
`name _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__logical_and, typename _Dom::value_type >::result_type > operator&& (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `const _ios_Fmtflags & operator&= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_ostate & operator&= (_ios_ostate &__a, _ios_ostate __b)`
- `const _ios_Openmode & operator&= (_ios_Openmode &__a, _ios_Openmode __b)`
- `constexpr byte & operator&= (byte &__l, byte __r) noexcept`
- `constexpr chars_format & operator&= (chars_format &__lhs, chars_format __rhs) noexcept`
- `launch & operator&= (launch &__x, launch __y) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__multiplies, typename _Dom::value_type >::result_type > operator* (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__multiplies, typename _Dom::value_type >::result_type > operator* (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`multiplies, typename _Dom1::value_type >::result_type > operator* (const _Expr< _Dom1, typename __`
`Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__multiplies, typename _Dom::value_type >::result_type > operator* (const typename __`
`Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__multiplies, typename _Dom::value_type >::result_type > operator* (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (_CharT __lhs, basic_string< _CharT, _Traits, _`
`_Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (_CharT __lhs, const basic_string< _CharT, _Traits,`
`_Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs,`
`_CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs,`
`basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs,`
`const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs,`
`const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const _CharT *__lhs, basic_string< _CharT, _`
`_Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const _CharT *__lhs, const basic_string< _CharT,`
`_Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _`
`_fun< struct std::__plus, typename _Dom::value_type >::result_type > operator+ (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _`
`_fun< struct std::__plus, typename _Dom::value_type >::result_type > operator+ (const _Expr< _Dom, type-`
`name _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__plus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__plus,`
`typename _Dom1::value_type >::result_type > operator+ (const _Expr< _Dom1, typename _Dom1::value_`
`type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &`
`__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc >`
`&__lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &`
`__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &`
`__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`

- `template<typename _Tp >`
`constexpr complex< _Tp > operator+ (const complex< _Tp > &__x)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > operator+ (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > operator+ (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Iterator >`
`constexpr move_iterator< _Iterator > operator+ (typename move_iterator< _Iterator >::difference_type __n, const move_iterator< _Iterator > &__x)`
- `template<typename _Iterator >`
`constexpr reverse_iterator< _Iterator > operator+ (typename reverse_iterator< _Iterator >::difference_type __n, const reverse_iterator< _Iterator > &__x)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > operator- (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > operator- (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__minus, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type >::result_type > operator- (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp >`
`constexpr complex< _Tp > operator- (const complex< _Tp > &__x)`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto operator- (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y) -> decltype(__x.base() - __y.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto operator- (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y) -> decltype(__y.base() - __x.base())`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > operator- (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > operator- (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > operator/ (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > operator/ (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__divides, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__divides, typename _Dom1::value_type >::result_type > operator/ (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__divides, typename _Dom::value_type >::result_type > operator/ (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__divides, typename _Dom::value_type >::result_type > operator/ (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _Tp, typename _Up, _Lock_policy _Lp>`
`bool operator< (const __shared_ptr< _Tp, _Lp > &__a, const __shared_ptr< _Up, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator< (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__less, typename _Dom::value_type >::result_type > operator< (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__less, typename _Dom::value_type >::result_type > operator< (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__less, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__less, typename _Dom1::value_type >::result_type > operator< (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator< (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_lt_t< _Up, _Tp >`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator< (const array< _Tp, _Nm > &__a, const array< _Tp, _Nm > &__b)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`

- `template<typename _Tp, typename _Alloc >`
`bool operator< (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator< (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Iterator >`
`constexpr bool operator< (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator< (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator< (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator< (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp >`
`constexpr bool operator< (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator< (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_lt_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator< (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_lt_t< _Tp, _Up >`
- `template<typename _Tp, typename _Seq >`
`bool operator< (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() > __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator< (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator< (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool operator< (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator< (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >__fun< struct std::__less, typename _Dom::value_type >::result_type > operator< (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool operator< (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator< (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >__fun< struct std::__less, typename _Dom::value_type >::result_type > operator< (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename... _Types>`
`constexpr bool operator< (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator< (monostate, monostate) noexcept`

- `template<typename _Tp >`
`constexpr bool operator< (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator< (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator< (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Ostream, typename _Tp >`
`__rvalue_stream_insertion_t< _Ostream, _Tp > operator<< (_Ostream &&__os, const _Tp &__x)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_money< _MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_time< _CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setbase __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setfill< _CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setprecision __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setw __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, basic_string_view< _CharT, _Traits > __str)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const complex< _Tp > &__x)`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > operator<< (byte __b, _IntegerType __shift) noexcept`
- `template<class _Dom >`
`__Expr< _BinClos< struct std::__shift_left, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__shift_left, typename _Dom::value_type >::result_type > operator<< (const __Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`__Expr< _BinClos< struct std::__shift_left, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__shift_left, typename _Dom::value_type >::result_type > operator<< (const __Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::_shift_left, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::_shift_left, typename _Dom1::value_type >::result_type > operator<< (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_shift_left, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_shift_left, typename _Dom::value_type >::result_type > operator<< (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_shift_left, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_shift_left, typename _Dom::value_type >::result_type > operator<< (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &, const std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &, const std::uniform_real_distribution< _RealType > &)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const binomial_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const chi_squared_distribution< _RealType > &__x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const discrete_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const gamma_distribution< _RealType > &__x)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const lognormal_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const negative_binomial_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const normal_distribution< _RealType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
poisson_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::bernoulli_distribution &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::geometric_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
student_t_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _IntegerType >
constexpr __byte_op_t< _IntegerType > & operator<= (byte &__b, _IntegerType __shift) noexcept`
- `template<typename _CharT, typename _Traits >
constexpr bool operator<= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<
_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >
constexpr bool operator<= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<
_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >
constexpr bool operator<= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >
__y) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>
bool operator<= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`

- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator<= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__less_equal, typename _Dom::value_type >::result_type > operator<= (const _Expr< _`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__less_equal, typename _Dom::value_type >::result_type > operator<= (const _Expr< _`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename _fun< struct std::`
`::__less_equal, typename _Dom1::value_type >::result_type > operator<= (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator<= (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_le_t< _Up, _Tp`
`>`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator<= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits,`
`_Alloc > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator<= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc`
`> &__y)`
- `template<typename _Iterator >`
`constexpr bool operator<= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool operator<= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator<= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _`
`_Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator<= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc >`
`&__y)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator<= (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_le_t< _Tp, _Up`
`>`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator<= (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_le_t<`
`_Tp, _Up >`

- `template<typename _Tp >`
`constexpr bool operator<= (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator<= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() } >= __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator<= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator<= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool operator<= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__less_equal, typename _Dom::value_type >::result_type > operator<= (const typename <`
`_Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool operator<= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator<= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__less_equal, typename _Dom::value_type >::result_type > operator<= (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename... _Types>`
`constexpr bool operator<= (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator<= (monostate, monostate) noexcept`
- `template<typename _Tp >`
`constexpr bool operator<= (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator<= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator<= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<three_way_comparable _Iterator>`
`constexpr compare_three_way_result_t< _Iterator, _Iterator > operator<= (const reverse_iterator< _Iterator`
`> &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, three_way_comparable_with< _IteratorL > _IteratorR>`
`constexpr compare_three_way_result_t< _IteratorL, _IteratorR > operator<= (const reverse_iterator< _`
`IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<`
`_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`

- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator==(const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator==(const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc>`
`bool operator==(const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom>`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__equal_to, typename _Dom::value_type >::result_type > operator==(const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom>`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__equal_to, typename _Dom::value_type >::result_type > operator==(const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2>`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__equal_to, typename _Dom1::value_type >::result_type > operator==(const _Expr< _Dom1, typename __`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up>`
`constexpr auto operator==(const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_eq_t< _Up, _Tp`
`>`
- `template<typename _T1, typename _T2>`
`constexpr bool operator==(const allocator< _T1 > &, const allocator< _T2 > &) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator==(const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _CharT>`
`constexpr __gnu_cxx::__enable_if< __is_char< _CharT >::value, bool >::__type operator==(const`
`basic_string< _CharT > &__lhs, const basic_string< _CharT > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc>`
`constexpr bool operator==(const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc>`
`constexpr bool operator==(const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT,`
`_Traits, _Alloc > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc>`
`bool operator==(const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool operator==(const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _StateT>`
`bool operator==(const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)`
- `template<typename _Res, typename... _Args>`
`bool operator==(const function< _Res(_Args...) > &__f, nullptr_t) noexcept`
- `template<typename _CharT, typename _Traits>`
`bool operator==(const istreambuf_iterator< _CharT, _Traits > &__a, const istreambuf_iterator< _CharT, _Traits`
`> &__b)`
- `template<typename _Tp, typename _Alloc>`
`bool operator==(const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool operator==(const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc`
`> &__y)`
- `template<typename _Bi_iter, typename _Alloc>`
`bool operator==(const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__`
`__m2)`

- `template<typename _Iterator >`
`constexpr bool operator== (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL , typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator== (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >`
`bool operator== (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool operator== (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp , typename _Up >`
`constexpr auto operator== (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp , typename _Up >`
`constexpr auto operator== (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool operator== (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp , typename _Seq >`
`bool operator== (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Iterator >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator== (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL , typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator== (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool operator== (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp , typename _Seq >`
`bool operator== (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename _RealType >`
`bool operator== (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__equal_to, typename _Dom::value_type >::result_type > operator== (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp , typename _Dp , typename _Up , typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool operator== (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp , typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator== (const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<class _Key , class _Tp , class _Hash , class _Pred , class _Alloc >`
`bool operator== (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key , class _Tp , class _Hash , class _Pred , class _Alloc >`
`bool operator== (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`

- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__equal_to, typename _Dom::value_type >::result_type > operator== (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename... _Types>`
`constexpr bool operator== (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`constexpr bool operator== (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator== (monostate, monostate) noexcept`
- `template<typename _Tp >`
`constexpr bool operator== (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator== (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool operator== (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator> (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator> (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__greater, typename _Dom::value_type >::result_type > operator> (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__greater, typename _Dom::value_type >::result_type > operator> (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__greater, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__greater, typename _Dom1::value_type >::result_type > operator> (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator> (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_gt_t< _Up, _Tp >`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator> (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`

- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator> (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Iterator >`
`constexpr bool operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator> (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator> (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator> (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator> (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool operator> (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator> (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator> (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool operator> (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::greater, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::greater, typename _Dom::value_type >::result_type > operator> (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool operator> (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator> (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__greater, typename _Dom::value_type >::result_type > operator> (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename... _Types>`
`constexpr bool operator> (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator> (monostate, monostate) noexcept`
- `template<typename _Tp >`
`constexpr bool operator> (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator> (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator> (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<`
`_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator>= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator>= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > operator>= (const <-`
`_Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > operator>= (const <-`
`_Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::<-`
`__greater_equal, typename _Dom1::value_type >::result_type > operator>= (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator>= (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_ge_t< _Up, _Tp`
`>`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool operator>= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits,`
`_Alloc > &__rhs) noexcept`

- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator>= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Iterator >`
`constexpr bool operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator>= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator>= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator>= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp >`
`constexpr bool operator>= (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator>= (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto operator>= (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename _Tp, typename _Seq >`
`bool operator>= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() <= __y.base() } -> convertible_to<bool>; }`
`constexpr bool operator>= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator>= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool operator>= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std:: __greater_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std:: __greater_equal, typename _Dom::value_type >::result_type > operator>= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`_GLIBCXX23_CONSTEXPR bool operator>= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`_GLIBCXX23_CONSTEXPR bool operator>= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std:: __greater_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std:: __greater_equal, typename _Dom::value_type >::result_type > operator>= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`

- `template<typename... _Types>`
`constexpr bool operator>= (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `constexpr bool operator>= (monostate, monostate) noexcept`
- `template<typename _Tp >`
`constexpr bool operator>= (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator>= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool operator>= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Istream, typename _Tp >`
`__rvalue_stream_extraction_t< _Istream, _Tp > operator>> (_Istream &&__is, _Tp &&__x)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Get_money< _↵ MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Get_time< _CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setbase __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setfill< _CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setprecision __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setw __f)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, basic_string< _↵ CharT, _Traits, _Alloc > &__str)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, complex< _Tp > &__x)`
- `template<>`
`basic_istream< char > & operator>> (basic_istream< char > &__is, basic_string< char > &__↵ __str)`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > operator>> (byte __b, _IntegerType __shift) noexcept`
- `template<class _Dom >`
`__Expr< _BinClos< struct std::__shift_right, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__shift_right, typename _Dom::value_type >::result_type > operator>> (const __Expr< _↵ Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`__Expr< _BinClos< struct std::__shift_right, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__shift_right, typename _Dom::value_type >::result_type > operator>> (const __Expr< _↵ Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1 , class _Dom2 >`
`_Expr< _BinClos< struct std::__shift_right, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__shift_right, typename _Dom1::value_type >::result_type > operator>> (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__shift_right, typename _Dom::value_type >::result_type > operator>> (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__shift_right, typename _Dom::value_type >::result_type > operator>> (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _IntType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &, std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &, std::uniform_real_distribution< _RealType > &)`
- `template<typename _IntType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, binomial_distribution< _IntType > &__x)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, chi_squared_distribution< _RealType > &__x)`
- `template<typename _RandomNumberEngine , size_t __p, size_t __r, typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _IntType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, discrete_distribution< _IntType > &__x)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, gamma_distribution< _RealType > &__x)`
- `template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, lognormal_distribution< _RealType > &__x)`
- `template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__x)`
- `template<typename _IntType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, negative_binomial_distribution< _IntType > &__x)`
- `template<typename _RealType , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, normal_distribution< _RealType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`poisson_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::bernoulli_distribution &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`student_t_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > & operator>>= (byte &__b, _IntegerType __shift) noexcept`
- `constexpr _los_Fmtflags operator^ (_los_Fmtflags __a, _los_Fmtflags __b)`
- `constexpr _los_lostate operator^ (_los_lostate __a, _los_lostate __b)`
- `constexpr _los_Openmode operator^ (_los_Openmode __a, _los_Openmode __b)`
- `constexpr byte operator^ (byte __l, byte __r) noexcept`
- `constexpr chars_format operator^ (chars_format __lhs, chars_format __rhs) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const _Expr< _↵`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__bitwise_xor, typename _Dom1::value_type >::result_type > operator^ (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr launch operator^ (launch __x, launch __y) noexcept`
- `const _los_Fmtflags & operator^= (_los_Fmtflags &__a, _los_Fmtflags __b)`
- `const _los_losestate & operator^= (_los_losestate &__a, _los_losestate __b)`
- `const _los_Openmode & operator^= (_los_Openmode &__a, _los_Openmode __b)`
- `constexpr byte & operator^= (byte &__l, byte __r) noexcept`
- `constexpr chars_format & operator^= (chars_format &__lhs, chars_format __rhs) noexcept`
- `launch & operator^= (launch &__x, launch __y) noexcept`
- `constexpr _los_Fmtflags operator| (_los_Fmtflags __a, _los_Fmtflags __b)`
- `constexpr _los_losestate operator| (_los_losestate __a, _los_losestate __b)`
- `constexpr _los_Openmode operator| (_los_Openmode __a, _los_Openmode __b)`
- `constexpr byte operator| (byte __l, byte __r) noexcept`
- `constexpr chars_format operator| (chars_format __lhs, chars_format __rhs) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > operator| (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > operator| (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__bitwise_or, typename _Dom1::value_type >::result_type > operator| (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > operator| (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > operator| (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr launch operator| (launch __x, launch __y) noexcept`
- `constexpr memory_order operator| (memory_order __m, __memory_order_modifier __mod)`
- `const _los_Fmtflags & operator|= (_los_Fmtflags &__a, _los_Fmtflags __b)`
- `const _los_losestate & operator|= (_los_losestate &__a, _los_losestate __b)`
- `const _los_Openmode & operator|= (_los_Openmode &__a, _los_Openmode __b)`
- `constexpr byte & operator|= (byte &__l, byte __r) noexcept`
- `constexpr chars_format & operator|= (chars_format &__lhs, chars_format __rhs) noexcept`
- `launch & operator|= (launch &__x, launch __y) noexcept`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__logical_or, typename _Dom::value_type >::result_type > operator|| (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__logical_or, typename _Dom::value_type >::result_type > operator|| (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _Expr, _Dom1, _Dom2 >, typename _fun< struct std::__`
`__logical_or, typename _Dom1::value_type >::result_type > operator|| (const _Expr< _Dom1, typename _`
`Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__logical_or, typename _Dom::value_type >::result_type > operator|| (const typename _`
`Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__logical_or, typename _Dom::value_type >::result_type > operator|| (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr _ios_Fmtflags operator~ (_ios_Fmtflags __a)`
- `constexpr _ios_ostate operator~ (_ios_ostate __a)`
- `constexpr _ios_Openmode operator~ (_ios_Openmode __a)`
- `constexpr byte operator~ (byte __b) noexcept`
- `constexpr chars_format operator~ (chars_format __fmt) noexcept`
- `constexpr launch operator~ (launch __x) noexcept`
- `template<typename _Tp >`
`optional (_Tp) -> optional< _Tp >`
- `template<typename _Fun, typename _Signature = typename __function_guide_helper<decltype(&_Fun::operator())>::type>`
`packaged_task (_Fun) -> packaged_task< _Signature >`
- `template<typename _Res, typename... _ArgTypes>`
`packaged_task (_Res*)(_ArgTypes...) -> packaged_task< _Res(_ArgTypes...)>`
- `template<typename _RAIter >`
`constexpr void partial_sort (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void partial_sort (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`constexpr void partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _Random`
`AccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _Random`
`AccessIterator __last, _Compare __comp)`
- `template<typename _Iter, typename _RAIter >`
`constexpr _RAIter partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter)`
- `template<typename _Iter, typename _RAIter, typename _Compare >`
`constexpr _RAIter partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter, _Compare)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator partial_sort_copy (_InputIterator __first, _InputIterator __last, _Random`
`AccessIterator __result_first, _RandomAccessIterator __result_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator partial_sort_copy (_InputIterator __first, _InputIterator __last, _Random`
`AccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`

- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _↵ BinaryOperation __binary_op)`
- `template<typename _BIter, typename _Predicate >`
`constexpr _BIter partition (_BIter, _BIter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Iter, typename _OIter1, typename _OIter2, typename _Predicate >`
`constexpr pair< _OIter1, _OIter2 > partition_copy (_Iter, _Iter, _OIter1, _OIter2, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`constexpr pair< _OutputIterator1, _OutputIterator2 > partition_copy (_InputIterator __first, _InputIterator __last, _OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _Filter, typename _Predicate >`
`constexpr _Filter partition_point (_Filter, _Filter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Tp >`
`complex< _Tp > polar (const _Tp &, const _Tp &=0)`
- `template<typename _RAIter >`
`constexpr void pop_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void pop_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`constexpr void pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp, int > popcount (_Tp __x) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type pow (_Tp __x, _Up __y)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _↵ Dom::value_type > pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom↵ ::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _↵ Dom::value_type > pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > pow (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2↵ ::value_type > &__e2)`
- `template<typename _Tp >`
`complex< _Tp > pow (const _Tp &, const complex< _Tp > &)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, const _Tp &)`
- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, const complex< _Tp > &)`

- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, int)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp > &__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp > &__x, const std::complex< _Up > &__y)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Pow, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > pow (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Pow, _Constant, _ValArray, _Tp, _Tp >, _Tp > pow (const typename valarray< _Tp >::value_type &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Pow, _ValArray, _Constant, _Tp, _Tp >, _Tp > pow (const valarray< _Tp > &__v, const typename valarray< _Tp >::value_type &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Pow, _ValArray, _ValArray, _Tp, _Tp >, _Tp > pow (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Pow, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > pow (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr float pow (float __x, float __y)`
- `constexpr long double pow (long double __x, long double __y)`
- `template<typename _BidirectionalIterator >`
`constexpr _BidirectionalIterator prev (_BidirectionalIterator __x, typename iterator_traits< _BidirectionalIterator >::difference_type __n=1)`
- `template<typename _BidirectionalIterator >`
`constexpr bool prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Blter >`
`constexpr bool prev_permutation (_Blter, _Blter)`
- `template<typename _Blter, typename _Compare >`
`constexpr bool prev_permutation (_Blter, _Blter, _Compare)`
- `template<typename _Compare, typename _Container, typename = _RequireNotAllocator< _Compare>, typename = _RequireNotAllocator< _Container>>>`
`priority_queue (_Compare, _Container) -> priority_queue< typename _Container::value_type, _Container, _Compare >`
- `template<typename _Compare, typename _Container, typename _Allocator, typename = _RequireNotAllocator< _Compare>, typename = _RequireNotAllocator< _Container>>>`
`priority_queue (_Compare, _Container, _Allocator) -> priority_queue< typename _Container::value_type, _Container, _Compare >`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits< _InputIterator >::value_type, typename _Compare = less< _ValT>, typename _Container = vector< _ValT>, typename = _RequireInputIter< _InputIterator>, typename = _RequireNotAllocator< _Compare>, typename = _RequireNotAllocator< _Container>>>`
`priority_queue (_InputIterator, _InputIterator, _Compare=_Compare(), _Container=_Container()) -> priority_queue< _ValT, _Container, _Compare >`
- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > proj (_Tp __x)`

- `template<typename _Tp >`
`std::complex< _Tp > proj (const std::complex< _Tp > &)`
- `template<typename _Arg, typename _Result >`
`pointer_to_unary_function< _Arg, _Result > ptr_fun (_Result(*__x)(_Arg))`
- `template<typename _Arg1, typename _Arg2, typename _Result >`
`pointer_to_binary_function< _Arg1, _Arg2, _Result > ptr_fun (_Result(*__x)(_Arg1, _Arg2))`
- `template<typename _RAIter >`
`constexpr void push_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void push_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`constexpr void push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _MoneyT >`
`_Put_money< _MoneyT > put_money (const _MoneyT &__mon, bool __intl=false)`
- `template<typename _CharT >`
`_Put_time< _CharT > put_time (const std::tm * __tmb, const _CharT * __fmt)`
- `template<typename _Container, typename = _RequireNotAllocator< _Container >>`
`queue (_Container) -> queue< typename _Container::value_type, _Container >`
- `template<typename _Container, typename _Allocator, typename = _RequireNotAllocator< _Container >>`
`queue (_Container, _Allocator) -> queue< typename _Container::value_type, _Container >`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto quoted (basic_string< _CharT, _Traits, _Alloc > &__string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits >`
`auto quoted (basic_string_view< _CharT, _Traits > __sv, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT >`
`auto quoted (const _CharT * __string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto quoted (const basic_string< _CharT, _Traits, _Alloc > &__string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _RAIter >`
`void random_shuffle (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Generator >`
`void random_shuffle (_RAIter, _RAIter, _Generator &&)`
- `template<typename _RandomAccessIterator >`
`void random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator &&__rand)`
- `template<typename _Container >`
`constexpr auto rbegin (_Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp, size_t _Nm >`
`constexpr reverse_iterator< _Tp * > rbegin (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto rbegin (const _Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp >`
`constexpr reverse_iterator< const _Tp * > rbegin (initializer_list< _Tp > __il) noexcept`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type real (_Tp __x)`

- `template<typename _Tp >`
`constexpr _Tp real (const complex< _Tp > &__z)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::value_type reduce (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp reduce (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Tp >`
`reference_wrapper (_Tp &) -> reference_wrapper< _Tp >`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > reinterpret_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Filter, typename _Tp >`
`constexpr _Filter remove (_Filter, _Filter, const _Tp &)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`constexpr _OIter remove_copy (_Iter, _Iter, _OIter, const _Tp &)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__value)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`constexpr _OIter remove_copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _Filter, typename _Predicate >`
`constexpr _Filter remove_if (_Filter, _Filter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Container >`
`constexpr auto rend (_Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp, size_t _Nm>`
`constexpr reverse_iterator< _Tp * > rend (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto rend (const _Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp >`
`constexpr reverse_iterator< const _Tp * > rend (initializer_list< _Tp > __il) noexcept`
- `template<typename _Filter, typename _Tp >`
`constexpr void replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`constexpr _OIter replace_copy (_Iter, _Iter, _OIter, const _Tp &, const _Tp &)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator replace_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`constexpr _OutputIterator replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _Iter, typename _OIter, typename _Predicate, typename _Tp >`
`constexpr _OIter replace_copy_if (_Iter, _Iter, _OIter, _Predicate, const _Tp &)`

- `template<typename _Filter, typename _Predicate, typename _Tp >`
`constexpr void replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`constexpr void replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp & __new_value)`
- `_Resetiosflags resetiosflags (ios_base::fmtflags __mask)`
- `void rethrow_exception (exception_ptr)`
- `template<typename _Ex >`
`void rethrow_if_nested (const _Ex & __ex)`
- `template<typename _Tp >`
`void return_temporary_buffer (_Tp * __p)`
- `template<typename _BidirectionalIterator >`
`constexpr void reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BIter >`
`constexpr void reverse (_BIter, _BIter)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`constexpr _OutputIterator reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`
- `template<typename _BIter, typename _OIter >`
`constexpr _OIter reverse_copy (_BIter, _BIter, _OIter)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type riemann_zeta (_Tp __s)`
- `float riemann_zetaf (float __s)`
- `long double riemann_zetal (long double __s)`
- `ios_base & right (ios_base & __base)`
- `template<typename _Filter >`
`constexpr _Filter rotate (_Filter, _Filter, _Filter)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last)`
- `template<typename _Filter, typename _OIter >`
`constexpr _OIter rotate_copy (_Filter, _Filter, _Filter, _OIter)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`constexpr _OutputIterator rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, _OutputIterator __result)`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > rotr (_Tp __x, int __s) noexcept`
- `template<typename _Tp >`
`constexpr _If_is_unsigned_integer< _Tp > rotr (_Tp __x, int __s) noexcept`
- `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomBitGenerator >`
`_SampleIterator sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomBitGenerator && __g)`
- `ios_base & scientific (ios_base & __base)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _ForwardIterator, typename _Searcher >`
`constexpr _ForwardIterator search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher & __searcher)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2)`

- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _Filter, typename _Size, typename _Tp >`
`constexpr _Filter search_n (_Filter, _Filter, _Size, const _Tp &)`
- `template<typename _Filter, typename _Size, typename _Tp, typename _BinaryPredicate >`
`constexpr _Filter search_n (_Filter, _Filter, _Size, const _Tp &, _BinaryPredicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`constexpr _ForwardIterator search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`constexpr _ForwardIterator search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`set (_InputIterator, _InputIterator, _Allocator) -> set< typename iterator_traits< _InputIterator >::value_type, less< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>>`
`set (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> set< typename iterator_traits< _InputIterator >::value_type, _Compare, _Allocator >`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`set (initializer_list< _Key >, _Allocator) -> set< _Key, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>>`
`set (initializer_list< _Key >, _Compare=_Compare(), _Allocator=_Allocator()) -> set< _Key, _Compare, _Allocator >`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`constexpr _OIter set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`constexpr _OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `new_handler set_new_handler (new_handler) throw ()`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`constexpr _OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `terminate_handler set_terminate (terminate_handler) noexcept`
- `unexpected_handler set_unexpected (unexpected_handler) noexcept`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`constexpr _OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `_Setbase setbase (int __base)`
- `template<typename _CharT >`
`_Setfill< _CharT > setfill (_CharT __c)`
- `_Setiosflags setiosflags (ios_base::fmtflags __mask)`
- `_Setprecision setprecision (int __n)`
- `_Setw setw (int __n)`
- `template<typename _Tp, typename _Del >`
`shared_ptr (unique_ptr< _Tp, _Del >) -> shared_ptr< _Tp >`
- `template<typename _Tp >`
`shared_ptr (weak_ptr< _Tp >) -> shared_ptr< _Tp >`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator shift_left (_ForwardIterator __first, _ForwardIterator __last, typename iterator_traits< _ForwardIterator >::difference_type __n)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator shift_right (_ForwardIterator __first, _ForwardIterator __last, typename iterator_traits< _ForwardIterator >::difference_type __n)`
- `ios_base & showbase (ios_base & __base)`
- `ios_base & showpoint (ios_base & __base)`
- `ios_base & showpos (ios_base & __base)`
- `template<typename _RAIter, typename _UGenerator >`
`void shuffle (_RAIter, _RAIter, _UGenerator &&)`
- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator && __g)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type sin (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Sin, _Expr, _Dom >, typename _Dom::value_type > sin (const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<typename _Tp >`
`complex< _Tp > sin (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Sin, _ValArray, _Tp >, _Tp > sin (const valarray< _Tp > & __v)`
- `constexpr float sin (float __x)`

- constexpr long double **sin** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **sinh** (_Tp __x)
- template<class _Dom >
_Expr< _UnClos< struct std::_Sinh, _Expr, _Dom >, typename _Dom::value_type > **sinh** (const _Expr< _Dom, typename _Dom::value_type > &__e)
- template<typename _Tp >
complex< _Tp > **sinh** (const **complex**< _Tp > &)
- template<typename _Tp >
_Expr< _UnClos< struct std::_Sinh, _ValArray, _Tp >, _Tp > **sinh** (const **valarray**< _Tp > &__v)
- constexpr float **sinh** (float __x)
- constexpr long double **sinh** (long double __x)
- template<typename _Container >
constexpr auto **size** (const _Container &__cont) noexcept(noexcept(__cont.size())) -> decltype(__cont.size())
- template<typename _Tp, size_t _Nm>
constexpr size_t **size** (const _Tp(&)[_Nm]) noexcept
- **ios_base** & **skipws** (**ios_base** & __base)
- template<typename _RAIter >
constexpr void **sort** (_RAIter, _RAIter)
- template<typename _RAIter, typename _Compare >
constexpr void **sort** (_RAIter, _RAIter, _Compare)
- template<typename _RandomAccessIterator >
constexpr void **sort** (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _RandomAccessIterator, typename _Compare >
constexpr void **sort** (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- template<typename _RAIter >
constexpr void **sort_heap** (_RAIter, _RAIter)
- template<typename _RAIter, typename _Compare >
constexpr void **sort_heap** (_RAIter, _RAIter, _Compare)
- template<typename _RandomAccessIterator >
constexpr void **sort_heap** (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _RandomAccessIterator, typename _Compare >
constexpr void **sort_heap** (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- template<contiguous_iterator _Iter, typename _End >
span (_Iter, _End) -> span< **remove_reference_t**< iter_reference_t< _Iter > > >
- template<ranges::contiguous_range _Range>
span (_Range &&) -> span< **remove_reference_t**< ranges::range_reference_t< _Range & > > >
- template<typename _Type, size_t _ArrayExtent>
span (_Type(&)[_ArrayExtent]) -> span< _Type, _ArrayExtent >
- template<typename _Type, size_t _ArrayExtent>
span (**array**< _Type, _ArrayExtent > &) -> span< _Type, _ArrayExtent >
- template<typename _Type, size_t _ArrayExtent>
span (const **array**< _Type, _ArrayExtent > &) -> span< const _Type, _ArrayExtent >
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **sph_bessel** (unsigned int __n, _Tp __x)
- float **sph_besself** (unsigned int __n, float __x)
- long double **sph_bessell** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **sph_legendre** (unsigned int __l, unsigned int __m, _Tp __theta)
- float **sph_legendref** (unsigned int __l, unsigned int __m, float __theta)
- long double **sph_legendrel** (unsigned int __l, unsigned int __m, long double __theta)

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_neumann (unsigned int __n, _Tp __x)`
- `float sph_neumannf (unsigned int __n, float __x)`
- `long double sph_neumannl (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type sqrt (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Sqrt, _Expr, _Dom >, typename _Dom::value_type > sqrt (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > sqrt (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Sqrt, _ValArray, _Tp >, _Tp > sqrt (const valarray< _Tp > &__v)`
- `constexpr float sqrt (float __x)`
- `constexpr long double sqrt (long double __x)`
- `template<typename _Container >`
`constexpr auto ssize (const _Container &__cont) noexcept(noexcept(__cont.size())) -> common_type_t<`
`ptrdiff_t, make_signed_t< decltype(__cont.size())> >`
- `template<typename _Tp, ptrdiff_t _Num>`
`constexpr ptrdiff_t ssize (const _Tp(&)[_Num]) noexcept`
- `template<typename _Blter, typename _Predicate >`
`_Blter stable_partition (_Blter, _Blter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void stable_sort (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`void stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Container, typename = _RequireNotAllocator< _Container >>`
`stack (_Container) -> stack< typename _Container::value_type, _Container >`
- `template<typename _Container, typename _Allocator, typename = _RequireNotAllocator< _Container >>`
`stack (_Container, _Allocator) -> stack< typename _Container::value_type, _Container >`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > static_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Callback >`
`stop_callback (stop_token, _Callback) -> stop_callback< _Callback >`
- `char * strchr (char *__s, int __n)`
- `char * strpbrk (char *__s1, const char *__s2)`
- `char * strrchr (char *__s, int __n)`
- `char * strstr (char *__s1, const char *__s2)`
- `template<typename _Tp, _Lock_policy _Lp>`
`void swap (__shared_ptr< _Tp, _Lp > &__a, __shared_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void swap (__weak_ptr< _Tp, _Lp > &__a, __weak_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`void swap (_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, _Rb_tree< _Key, _Val, _KeyOfValue,`
`_Compare, _Alloc > &__y)`

- `template<typename _Tp >`
`constexpr Require<__not_<__is_tuple_like<_Tp > >, is_move_constructible<_Tp >, is_move_assignable<`
`_Tp > > swap (_Tp &, _Tp &) noexcept(__and_< is_nothrow_move_constructible<_Tp >, is_nothrow_move_assignable<`
`_Tp > >::value)`
- `template<typename _Tp >`
`constexpr enable_if< __and_< __not_< __is_tuple_like<_Tp > >, is_move_constructible<_Tp >, is_move_assignable<`
`_Tp > >::value >::type swap (_Tp &__a, _Tp &__b) noexcept(/*conditional */)`
`is_nothrow_move_assignable<_Tp > >`
- `template<typename _Tp, size_t _Nm>`
`constexpr enable_if< __is_swappable<_Tp >::value >::type swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])`
`noexcept(/*conditional */)`
- `template<typename _Tp, size_t _Nm>`
`constexpr __enable_if_t< __is_swappable<_Tp >::value > swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])`
`noexcept(__is_nothrow_swappable<_Tp >::value)`
- `void swap (any &__x, any &__y) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`enable_if<!__array_traits<_Tp, _Nm >::is_swappable::value >::type swap (array<_Tp, _Nm > &, array<`
`_Tp, _Nm > &)=delete`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr enable_if< __array_traits<_Tp, _Nm >::is_swappable::value >::type swap (array<_Tp, _Nm >`
`&__one, array<_Tp, _Nm > &__two) noexcept(noexcept(__one.swap(__two)))`
- `template<class _CharT, class _Traits >`
`void swap (basic_filebuf<_CharT, _Traits > &__x, basic_filebuf<_CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_fstream<_CharT, _Traits > &__x, basic_fstream<_CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_ifstream<_CharT, _Traits > &__x, basic_ifstream<_CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_istream<_CharT, _Traits, _Allocator > &__x, basic_istream<_CharT, _Traits, _`
`Allocator > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_ofstream<_CharT, _Traits > &__x, basic_ofstream<_CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_ostringstream<_CharT, _Traits, _Allocator > &__x, basic_ostringstream<_CharT, _Traits,`
`_Allocator > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr void swap (basic_string<_CharT, _Traits, _Alloc > &__lhs, basic_string<_CharT, _Traits, _Alloc >`
`&__rhs) noexcept(/*conditional */)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_stringbuf<_CharT, _Traits, _Allocator > &__x, basic_stringbuf<_CharT, _Traits, _Allocator >`
`&__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_stringstream<_CharT, _Traits, _Allocator > &__x, basic_stringstream<_CharT, _Traits, _`
`Allocator > &__y)`
- `template<typename _Tp, typename _Alloc >`
`void swap (deque<_Tp, _Alloc > &__x, deque<_Tp, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Tp, typename _Alloc >`
`void swap (forward_list<_Tp, _Alloc > &__lx, forward_list<_Tp, _Alloc > &__ly) noexcept(noexcept(__lx.<`
`swap(__ly)))`
- `template<typename _Res, typename... _Args>`
`void swap (function<_Res(_Args...)> &__x, function<_Res(_Args...)> &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`void swap (list<_Tp, _Alloc > &__x, list<_Tp, _Alloc > &__y) noexcept(/*conditional */)`

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void swap (map< _Key, _Tp, _Compare, _Alloc > &__x, map< _Key, _Tp, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Bi_iter, typename _Alloc >`
`void swap (match_results< _Bi_iter, _Alloc > &__lhs, match_results< _Bi_iter, _Alloc > &__rhs) noexcept`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void swap (multimap< _Key, _Tp, _Compare, _Alloc > &__x, multimap< _Key, _Tp, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void swap (multiset< _Key, _Compare, _Alloc > &__x, multiset< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Tp >`
`enable_if_t<!(is_move_constructible_v< _Tp > &&is_swappable_v< _Tp >)> swap (optional< _Tp > &, optional< _Tp > &)=delete`
- `template<typename _Tp >`
`constexpr enable_if_t< is_move_constructible_v< _Tp > &&is_swappable_v< _Tp > > swap (optional< _Tp > &__lhs, optional< _Tp > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))`
- `template<typename _Res, typename... _ArgTypes>`
`void swap (packaged_task< _Res(_ArgTypes...) > &__x, packaged_task< _Res(_ArgTypes...) > &__y) noexcept`
- `template<typename _Tp, typename _Sequence, typename _Compare >`
`enable_if< __and< __is_swappable< _Sequence >, __is_swappable< _Compare > >::value >::type swap (priority_queue< _Tp, _Sequence, _Compare > &__x, priority_queue< _Tp, _Sequence, _Compare > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Res >`
`void swap (promise< _Res > &__x, promise< _Res > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`enable_if< __is_swappable< _Seq >::value >::type swap (queue< _Tp, _Seq > &__x, queue< _Tp, _Seq > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void swap (set< _Key, _Compare, _Alloc > &__x, set< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Tp, typename _Seq >`
`enable_if< __is_swappable< _Seq >::value >::type swap (stack< _Tp, _Seq > &__x, stack< _Tp, _Seq > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`constexpr enable_if<! __and< __is_swappable< _Elements >... >::value >::type swap (tuple< _Elements... > &, tuple< _Elements... > &)=delete`
- `template<typename... _Elements>`
`constexpr enable_if< __and< __is_swappable< _Elements >... >::value >::type swap (tuple< _Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Dp >`
`enable_if<! __is_swappable< _Dp >::value >::type swap (unique_ptr< _Tp, _Dp > &, unique_ptr< _Tp, _Dp > &)=delete`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`

- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Types>`
`enable_if_t<!(is_move_constructible_v< _Types > &&...) &&(is_swappable_v< _Types > &&...)> swap`
`(variant< _Types... > &, variant< _Types... > &)=delete`
- `template<typename... _Types>`
`constexpr enable_if_t<(is_move_constructible_v< _Types > &&...) &&(is_swappable_v< _Types > &&...)> swap`
`(variant< _Types... > &__lhs, variant< _Types... > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))`
- `template<typename _Tp, typename _Alloc >`
`constexpr void swap (vector< _Tp, _Alloc > &__x, vector< _Tp, _Alloc > &__y) noexcept(*conditional *)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter2 swap_ranges (_Filter1, _Filter1, _Filter2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator2 swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `const error_category & system_category () noexcept`
- `template<typename _Tp >`
`constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type tan (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Tan, _Expr, _Dom >, typename _Dom::value_type > tan (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > tan (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Tan, _ValArray, _Tp >, _Tp > tan (const valarray< _Tp > &__v)`
- `constexpr float tan (float __x)`
- `constexpr long double tan (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type tanh (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Tanh, _Expr, _Dom >, typename _Dom::value_type > tanh (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > tanh (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Tanh, _ValArray, _Tp >, _Tp > tanh (const valarray< _Tp > &__v)`
- `constexpr float tanh (float __x)`
- `constexpr long double tanh (long double __x)`
- `void terminate () noexcept`
- `template<typename _Tp >`
`void throw_with_nested (_Tp &&__t)`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &... > tie (_Elements &... __args) noexcept`
- `template<typename _Tp >`
`constexpr _Tp * to_address (_Tp *__ptr) noexcept`
- `template<typename _Ptr >`
`constexpr auto to_address (const _Ptr &__ptr) noexcept`
- `template<typename _Tp, size_t _Nm>`
`constexpr array< remove_cv_t< _Tp >, _Nm > to_array (_Tp(&&__a)[_Nm]) noexcept(is_nothrow_move_constructible_v< _Tp >)`

- `template<typename _Tp, size_t _Nm>`
`constexpr array< remove_cv_t< _Tp >, _Nm > to_array (_Tp(&__a)[_Nm]) noexcept(is_nothrow_↵`
`constructible_v< _Tp, _Tp & >)`
- `to_chars_result to_chars (char *, char *, bool, int=10)=delete`
- `to_chars_result to_chars (char * __first, char * __last, char __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, signed char __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, signed int __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, signed long __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, signed long long __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, signed short __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, unsigned char __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, unsigned int __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, unsigned long __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, unsigned long long __value, int __base=10)`
- `to_chars_result to_chars (char * __first, char * __last, unsigned short __value, int __base=10)`
- `template<typename _IntegerType >`
`constexpr _IntegerType to_integer (__byte_op_t< _IntegerType > __b) noexcept`
- `string to_string (int __val)`
- `string to_string (long __val)`
- `string to_string (long long __val)`
- `string to_string (unsigned __val)`
- `string to_string (unsigned long __val)`
- `string to_string (unsigned long long __val)`
- `template<typename _CharT >`
`_CharT tolower (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`_CharT toupper (_CharT __c, const locale & __loc)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`constexpr _OIter transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BinaryOperation >`
`constexpr _OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BinaryOperation)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`constexpr _OutputIterator transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, ↵`
`_UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵`
`_OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator transform_exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator`
`__result, _Tp __init, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator`
`__result, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation, typename _Tp >`
`constexpr _OutputIterator transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator`
`__result, _BinaryOperation __binary_op, _UnaryOperation __unary_op, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _Tp transform_reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation ↵`
`__binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp`
`__init)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2>`
`>`
`constexpr _Tp transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp`
`__init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`
- `template<typename _L1, typename _L2, typename... _L3>`
`int try_lock (_L1 &__l1, _L2 &__l2, _L3 &... __l3)`
- `template<typename... _UTypes>`
`tuple (_UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename... _UTypes>`
`tuple (allocator_arg_t, _Alloc, _UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename _T1, typename _T2 >`
`tuple (allocator_arg_t, _Alloc, pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename _Alloc, typename... _UTypes>`
`tuple (allocator_arg_t, _Alloc, tuple< _UTypes... >) -> tuple< _UTypes... >`
- `template<typename _T1, typename _T2 >`
`tuple (pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename... _Tpls, typename = typename enable_if<__and<__is_tuple_like<_Tpls>...>::value>::type>`
`constexpr auto tuple_cat (_Tpls &&... __tpls) -> typename __tuple_cat_result< _Tpls... >::type`
- `bool uncaught_exception () noexcept`
- `int uncaught_exceptions () noexcept`
- `void undeclare_no_pointers (char *, size_t)`
- `template<typename _Tp >`
`_Tp * undeclare_reachable (_Tp *__p)`
- `void unexpected ()`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp * uninitialized_construct_using_allocator (_Tp *__p, const _Alloc &__a, _Args &&... __args)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator uninitialized_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator uninitialized_copy_n (_InputIterator __first, _Size __n, _ForwardIterator __result)`
- `template<typename _ForwardIterator >`
`void uninitialized_default_construct (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator uninitialized_default_construct_n (_ForwardIterator __first, _Size __count)`
- `template<typename _ForwardIterator, typename _Tp >`
`void uninitialized_fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp >`
`_ForwardIterator uninitialized_fill_n (_ForwardIterator __first, _Size __n, const _Tp &__x)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator uninitialized_move (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`pair< _InputIterator, _ForwardIterator > uninitialized_move_n (_InputIterator __first, _Size __count, _ForwardIterator __result)`
- `template<typename _ForwardIterator >`
`void uninitialized_value_construct (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator uninitialized_value_construct_n (_ForwardIterator __first, _Size __count)`
- `template<typename _Filter >`
`constexpr _Filter unique (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`constexpr _Filter unique (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator unique (_ForwardIterator __first, _ForwardIterator __last)`

- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __↵`
`binary_pred)`
- `template<typename _Iter, typename _OIter >`
`constexpr _OIter unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _BinaryPredicate >`
`constexpr _OIter unique_copy (_Iter, _Iter, _OIter, _BinaryPredicate)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, ↵`
`_BinaryPredicate __binary_pred)`
- `ios_base & unitbuf (ios_base & __base)`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator<_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, _Allocator) -> unordered_map< __iter_key_t< _InputIterator >,`
`__iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_key_t< _InputIterator`
`> >, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator<_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, typename unordered_map< int, int >::size_type, _Allocator) ->`
`unordered_map< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< Input↵`
`Iterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, typename unordered_map< int, int >::size_type, _Hash, ↵`
`_Allocator) -> unordered_map< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Hash,`
`equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash< __iter_key_t< _InputIterator>>, typename _Pred = equal_to< __iter_key_↵`
`t< _InputIterator>>, typename _Allocator = allocator< __iter_to_alloc_t< _InputIterator>>, typename = _RequireInputIter<_InputIterator>,`
`typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator< ↵`
`_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, typename unordered_map< int, int >::size_type={}, _Hash= ↵`
`Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map< __iter_key_t< _InputIterator >, __iter_↵`
`val_t< _InputIterator >, _Hash, _Pred, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`unordered_map (initializer_list< pair< _Key, _Tp > >, _Allocator) -> unordered_map< _Key, _Tp, hash< _Key`
`>, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type, ↵`
`_Allocator) -> unordered_map< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>,`
`typename = _RequireAllocator<_Allocator>>>`
`unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type, _Hash,`
`_Allocator) -> unordered_map< _Key, _Tp, _Hash, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash = hash< _Key>, typename _Pred = equal_to< _Key>, typename _Allocator`
`= allocator< pair< const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>,`
`typename = _RequireAllocator<_Allocator>>>`
`unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type={},`
`_Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map< _Key, _Tp, _Hash, _Pred, ↵`
`_Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator<_Allocator>>>`

- unordered_multimap** (*_InputIterator*, *_InputIterator*, *_Allocator*) -> **unordered_multimap**< *__iter_key_t*< *__iter_val_t*< *_InputIterator* >, *__iter_val_t*< *_InputIterator* >, *hash*< *__iter_key_t*< *_InputIterator* >, *equal_to*< *__iter_key_t*< *_InputIterator* >, *_Allocator* >
- template<typename *_InputIterator* , typename *_Allocator* , typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*_InputIterator*, *_InputIterator*, **unordered_multimap**< int, int >::size_type, *_Allocator*) -> **unordered_multimap**< *__iter_key_t*< *_InputIterator* >, *__iter_val_t*< *_InputIterator* >, *hash*< *__iter_key_t*< *_InputIterator* >, *equal_to*< *__iter_key_t*< *_InputIterator* >, *_Allocator* >
 - template<typename *_InputIterator* , typename *_Hash* , typename *_Allocator* , typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*_InputIterator*, *_InputIterator*, **unordered_multimap**< int, int >::size_type, *_Hash*, *_Allocator*) -> **unordered_multimap**< *__iter_key_t*< *_InputIterator* >, *__iter_val_t*< *_InputIterator* >, *_Hash*, *equal_to*< *__iter_key_t*< *_InputIterator* >, *_Allocator* >
 - template<typename *_InputIterator* , typename *_Hash* = *hash*<*__iter_key_t*<*_InputIterator*>>, typename *_Pred* = *equal_to*<*__iter_key_t*<*_InputIterator*>>, typename *_Allocator* = *allocator*<*__iter_to_alloc_t*<*_InputIterator*>>, typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireNotAllocator*<*_Pred*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*_InputIterator*, *_InputIterator*, **unordered_multimap**< int, int >::size_type={}, *_Hash*=*_Hash*(), *_Pred*=*_Pred*(), *_Allocator*=*_Allocator*()) -> **unordered_multimap**< *__iter_key_t*< *_InputIterator* >, *__iter_val_t*< *_InputIterator* >, *_Hash*, *_Pred*, *_Allocator* >
 - template<typename *_Key* , typename *_Tp* , typename *_Allocator* , typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*initializer_list*< *pair*< *_Key*, *_Tp* > >, *_Allocator*) -> **unordered_multimap**< *_Key*, *_Tp*, *hash*< *_Key* >, *equal_to*< *_Key* >, *_Allocator* >
 - template<typename *_Key* , typename *_Tp* , typename *_Allocator* , typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*initializer_list*< *pair*< *_Key*, *_Tp* > >, **unordered_multimap**< int, int >::size_type, *_Allocator*) -> **unordered_multimap**< *_Key*, *_Tp*, *hash*< *_Key* >, *equal_to*< *_Key* >, *_Allocator* >
 - template<typename *_Key* , typename *_Tp* , typename *_Hash* , typename *_Allocator* , typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*initializer_list*< *pair*< *_Key*, *_Tp* > >, **unordered_multimap**< int, int >::size_type, *_Hash*, *_Allocator*) -> **unordered_multimap**< *_Key*, *_Tp*, *_Hash*, *equal_to*< *_Key* >, *_Allocator* >
 - template<typename *_Key* , typename *_Tp* , typename *_Hash* = *hash*<*_Key*>, typename *_Pred* = *equal_to*<*_Key*>, typename *_Allocator* = *allocator*<*pair*< *const* *_Key*, *_Tp* >>, typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireNotAllocator*<*_Pred*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multimap (*initializer_list*< *pair*< *_Key*, *_Tp* > >, **unordered_multimap**< int, int >::size_type={}, *_Hash*=*_Hash*(), *_Pred*=*_Pred*(), *_Allocator*=*_Allocator*()) -> **unordered_multimap**< *_Key*, *_Tp*, *_Hash*, *_Pred*, *_Allocator* >
 - template<typename *_InputIterator* , typename *_Allocator* , typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multiset (*_InputIterator*, *_InputIterator*, **unordered_multiset**< int >::size_type, *_Allocator*) -> **unordered_multiset**< typename *iterator_traits*< *_InputIterator* >::value_type, *hash*< typename *iterator_traits*< *_InputIterator* >::value_type >, *equal_to*< typename *iterator_traits*< *_InputIterator* >::value_type >, *_Allocator* >
 - template<typename *_InputIterator* , typename *_Hash* , typename *_Allocator* , typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multiset (*_InputIterator*, *_InputIterator*, **unordered_multiset**< int >::size_type, *_Hash*, *_Allocator*) -> **unordered_multiset**< typename *iterator_traits*< *_InputIterator* >::value_type, *_Hash*, *equal_to*< typename *iterator_traits*< *_InputIterator* >::value_type >, *_Allocator* >
 - template<typename *_InputIterator* , typename *_Hash* = *hash*<typename *iterator_traits*<*_InputIterator*>::value_type>, typename *_Pred* = *equal_to*<typename *iterator_traits*<*_InputIterator*>::value_type>, typename *_Allocator* = *allocator*<typename *iterator_traits*<*_InputIterator*>::value_type>, typename = *_RequireInputIter*<*_InputIterator*>, typename = *_RequireNotAllocatorOrIntegral*<*_Hash*>, typename = *_RequireNotAllocator*<*_Pred*>, typename = *_RequireAllocator*<*_Allocator*>>>
unordered_multiset (*_InputIterator*, *_InputIterator*, **unordered_multiset**< int >::size_type={}, *_Hash*=*_Hash*(), *_Pred*=*_Pred*(), *_Allocator*=*_Allocator*()) -> **unordered_multiset**< typename *iterator_traits*< *_InputIterator* >::value_type, *_Hash*, *_Pred*, *_Allocator* >

- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type, _Allocator) -> `unordered_multiset`< _Tp, `hash`< _Tp >, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type, _Hash, _Allocator) -> `unordered_multiset`< _Tp, _Hash, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_multiset`< _Tp, _Hash, _Pred, _Allocator >
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
unordered_set (_InputIterator, _InputIterator, `unordered_set`< int >::size_type, _Allocator) -> `unordered_set`< typename `iterator_traits`< _InputIterator >::value_type, `hash`< typename `iterator_traits`< _InputIterator >::value_type >, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_set (_InputIterator, _InputIterator, `unordered_set`< int >::size_type, _Hash, _Allocator) -> `unordered_set`< typename `iterator_traits`< _InputIterator >::value_type, _Hash, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >
- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_set (_InputIterator, _InputIterator, `unordered_set`< int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_set`< typename `iterator_traits`< _InputIterator >::value_type, _Hash, _Pred, _Allocator >
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
unordered_set (`initializer_list`< _Tp >, `unordered_set`< int >::size_type, _Allocator) -> `unordered_set`< _Tp, `hash`< _Tp >, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_set (`initializer_list`< _Tp >, `unordered_set`< int >::size_type, _Hash, _Allocator) -> `unordered_set`< _Tp, _Hash, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_set (`initializer_list`< _Tp >, `unordered_set`< int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_set`< _Tp, _Hash, _Pred, _Allocator >
- `template<typename _Filter, typename _Tp >`
constexpr _Filter upper_bound (_Filter, _Filter, const _Tp &)
- `template<typename _Filter, typename _Tp, typename _Compare >`
constexpr _Filter upper_bound (_Filter, _Filter, const _Tp &, _Compare)
- `template<typename _ForwardIterator, typename _Tp >`
constexpr _ForwardIterator upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
constexpr _ForwardIterator upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)
- `ios_base & uppercase` (`ios_base` & __base)
- `template<typename _Facet >`
const _Facet & use_facet (const `locale` & __loc)

- `template<_Std_pair _Tp, typename _Alloc >`
`constexpr auto uses_allocator_construction_args (const _Alloc &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto uses_allocator_construction_args (const _Alloc &, _Up &&, _Vp &&) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto uses_allocator_construction_args (const _Alloc &, const pair< _Up, _Vp > &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto uses_allocator_construction_args (const _Alloc &, pair< _Up, _Vp > &&) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`requires (! _Std_pair<_Tp>)`
`constexpr auto uses_allocator_construction_args (const _Alloc &__a, _Args &&... __args) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Tuple1, typename _Tuple2 >`
`constexpr auto uses_allocator_construction_args (const _Alloc &__a, piecewise_construct_t, _Tuple1 &&__x, _Tuple2 &&__y) noexcept`
- `template<typename _Tp, size_t _Nm>`
`valarray (const _Tp(&)[_Nm], size_t) -> valarray< _Tp >`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`vector (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> vector< _ValT, _Allocator >`
- `template<typename _Visitor, typename... _Variants>`
`constexpr __detail::__variant::__visit_result_t< _Visitor, _Variants... > visit (_Visitor &&, _Variants &&...)`
- `template<typename _Visitor, typename... _Variants>`
`constexpr __detail::__variant::__visit_result_t< _Visitor, _Variants... > visit (_Visitor &&__visitor, _Variants &&... __variants)`
- `template<typename _Res, typename _Visitor, typename... _Variants>`
`constexpr _Res visit (_Visitor &&__visitor, _Variants &&... __variants)`
- `wchar_t * wcschr (wchar_t * __p, wchar_t __c)`
- `wchar_t * wcspbrk (wchar_t * __s1, const wchar_t * __s2)`
- `wchar_t * wcsrchr (wchar_t * __p, wchar_t __c)`
- `wchar_t * wcsstr (wchar_t * __s1, const wchar_t * __s2)`
- `template<typename _Tp >`
`weak_ptr (shared_ptr< _Tp >) -> weak_ptr< _Tp >`
- `wchar_t * wmemchr (wchar_t * __p, wchar_t __c, size_t __n)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & ws (basic_istream< _CharT, _Traits > &__is)`

- `template<typename _ValueType >`
`_ValueType any_cast (any &__any)`
- `template<typename _ValueType >`
`_ValueType any_cast (any &&__any)`

- `template<typename _ValueType >`
`const _ValueType * any_cast (const any * __any) noexcept`
- `template<typename _ValueType >`
`_ValueType * any_cast (any * __any) noexcept`

- `template<size_t _Nb>`
`bitset< _Nb > operator& (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`

- `template<size_t _Nb>`
`bitset<_Nb> operator| (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<size_t _Nb>`
`bitset<_Nb> operator^ (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_istream<_CharT, _Traits> & operator>> (std::basic_istream<_CharT, _Traits> &__is, bitset<_Nb> &__x)`
- `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_ostream<_CharT, _Traits> & operator<< (std::basic_ostream<_CharT, _Traits> &__os, const bitset<_Nb> &__x)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator+ (const complex<_Tp> &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator+ (const complex<_Tp> &__x, const _Tp &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator+ (const _Tp &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator- (const complex<_Tp> &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator- (const complex<_Tp> &__x, const _Tp &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator- (const _Tp &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator* (const complex<_Tp> &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator* (const complex<_Tp> &__x, const _Tp &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator* (const _Tp &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator/ (const complex<_Tp> &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator/ (const complex<_Tp> &__x, const _Tp &__y)`
- `template<typename _Tp>`
`constexpr complex<_Tp> operator/ (const _Tp &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr bool operator== (const complex<_Tp> &__x, const complex<_Tp> &__y)`
- `template<typename _Tp>`
`constexpr bool operator== (const complex<_Tp> &__x, const _Tp &__y)`

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, unsigned char &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, signed char &__c)`

- `template<typename _CharT, typename _Traits, size_t _Num>`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__in, _CharT(&__s)[_Num])`
- `template<class _Traits, size_t _Num>`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, unsigned char(&__s)[_Num])`
- `template<class _Traits, size_t _Num>`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, signed char(&__s)[_Num])`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, _CharT __c)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, char __c)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, char __c)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, signed char __c)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, unsigned char __c)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, wchar_t)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, char16_t)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, char32_t)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & operator<< (basic_ostream< wchar_t, _Traits > &, char16_t)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & operator<< (basic_ostream< wchar_t, _Traits > &, char32_t)=delete`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, const _CharT *__s)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, const char *__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const char *__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const signed char *__s)`

- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const unsigned char`
`*__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, const wchar_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, const char16_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &, const char32_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & operator<< (basic_ostream< wchar_t, _Traits > &, const char16_t *`
`t *)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & operator<< (basic_ostream< wchar_t, _Traits > &, const char32_t *`
`t *)=delete`

Matching, Searching, and Replacing

- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Alloc, typename _Rx_traits >`
`bool regex_match (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const`
`basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_t`
`_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,`
`_Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_type, class _Rx_traits >`
`bool regex_match (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type`
`__f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Str_allocator > &__s, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool regex_search (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type`
`__f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > &__s, const basic_regex<`
`_Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __flags=regex_constants::match_default)`

- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_`
`_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,`
`_Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter __regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const _Ch_type * __fmt, size_t __len, regex_constants::match_flag_type __flags)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _`
`_Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type`
`__flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _`
`Rx_traits > &__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s,`
`const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _Fst, _Fsa > &__fmt,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type, _St, _Sa > regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type * __fmt, regex_constants::match_flag_type __`
`flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type > regex_replace (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_`
`traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type __`
`flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type >`
`basic_string< _Ch_type > regex_replace (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits >`
`&__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & get (pair< _Tp1, _Tp2 > &__in) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && get (pair< _Tp1, _Tp2 > &&__in) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & get (const pair< _Tp1, _Tp2 > &__in)`
`noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && get (const pair< _Tp1, _Tp2 > &&__in)`
`noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & get (pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & get (const pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && get (pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp && get (const pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & get (pair< _Up, _Tp > &__p) noexcept`

- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & get (const pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && get (pair< _Up, _Tp > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp && get (const pair< _Up, _Tp > &&__p) noexcept`

Variables

- `template<typename _Tp >`
`constexpr auto __denorm_min_v`
- `template<typename _Tp >`
`constexpr auto __digits10_v`
- `template<typename _Tp >`
`constexpr auto __digits_v`
- `template<typename _Tp >`
`constexpr auto __epsilon_v`
- `template<typename _Tp >`
`constexpr auto __finite_max_v`
- `template<typename _Tp >`
`constexpr auto __finite_min_v`
- `template<template< typename > class _Trait, typename _Tp >`
`constexpr bool __has_iec559_behavior_v`
- `template<typename _Tp >`
`constexpr bool __has_iec559_storage_format_v`
- `template<typename _Tp >`
`constexpr auto __infinity_v`
- `static ios_base::Init __ioint`
- `template<typename >`
`constexpr bool __is_in_place_type_v`
- `template<typename _Tp >`
`constexpr bool __is_in_place_type_v< in_place_type_t< _Tp > >`
- `template<typename _Tp >`
`constexpr bool __is_optional_v`
- `template<typename _Tp >`
`constexpr bool __is_optional_v< optional< _Tp > >`
- `template<typename _Tp >`
`constexpr bool __is_pair`
- `template<typename _Tp, typename _Up >`
`constexpr bool __is_pair< const pair< _Tp, _Up > >`
- `template<typename _Tp, typename _Up >`
`constexpr bool __is_pair< pair< _Tp, _Up > >`
- `template<typename _Up >`
`static constexpr bool __is_shared_ptr`
- `template<typename _Up >`
`static constexpr bool __is_shared_ptr< shared_ptr< _Up > >`
- `template<typename _Tp >`
`constexpr auto __max_digits10_v`
- `template<typename _Tp >`
`constexpr auto __max_exponent10_v`
- `template<typename _Tp >`
`constexpr auto __max_exponent_v`

- `template<typename _Tp >`
`constexpr auto __min_exponent10_v`
- `template<typename _Tp >`
`constexpr auto __min_exponent_v`
- `template<typename _Tp >`
`constexpr auto __norm_min_v`
- `template<typename _Tp >`
`constexpr bool __platform_wait_uses_type`
- `template<typename _Tp >`
`constexpr auto __quiet_NaN_v`
- `template<typename _Tp >`
`constexpr auto __radix_v`
- `template<typename _Tp >`
`constexpr auto __reciprocal_overflow_threshold_v`
- `template<typename _Tp >`
`constexpr auto __round_error_v`
- `template<typename _Tp >`
`constexpr auto __signaling_NaN_v`
- `template<template< typename... > class _Trait, typename _Tp, typename _Tuple >`
`constexpr bool __unpack_std_tuple`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool __unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > &`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool __unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool __unpack_std_tuple< _Trait, _Tp, tuple< _Up... > &`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool __unpack_std_tuple< _Trait, _Tp, tuple< _Up... > >`
- `template<template< typename > class _Trait, typename _Tp >`
`constexpr bool __value_exists_v`
- `constexpr adopt_lock_t adopt_lock`
- `constexpr __cmp_cust::Partial_fallback compare_partial_order_fallback`
- `constexpr __cmp_cust::Strong_fallback compare_strong_order_fallback`
- `constexpr __cmp_cust::Weak_fallback compare_weak_order_fallback`
- `constexpr default_sentinel_t default_sentinel`
- `constexpr defer_lock_t defer_lock`
- `constexpr destroying_delete_t destroying_delete`
- `template<typename _Sent, typename _Iter >`
`constexpr bool disable_sized_sentinel_for`
- `template<typename _Iterator1, typename _Iterator2 >`
`constexpr bool disable_sized_sentinel_for< reverse_iterator< _Iterator1 >, reverse_iterator< _Iterator2 >`
`>`
- `constexpr size_t dynamic_extent`
- `template<typename _Tp >`
`constexpr bool has_unique_object_representations_v`
- `constexpr _Swallow_assign ignore`
- `constexpr in_place_t in_place`
- `template<size_t _Idx>`
`constexpr in_place_index_t< _Idx > in_place_index`
- `template<typename _Tp >`
`constexpr in_place_type_t< _Tp > in_place_type`

- `template<typename _Tp >`
`constexpr bool is_aggregate_v`
- `template<typename _Tp >`
`constexpr bool is_bind_expression_v`
- `template<typename _Tp >`
`constexpr bool is_bounded_array_v`
- `template<typename _Tp >`
`constexpr bool is_error_code_enum_v`
- `template<typename _Tp >`
`constexpr bool is_error_condition_enum_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_layout_compatible_v`
- `template<typename _From, typename _To >`
`constexpr bool is_nothrow_convertible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_swappable_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_nothrow_swappable_with_v`
- `template<typename _Tp >`
`constexpr int is_placeholder_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool is_pointer_interconvertible_base_of_v`
- `template<typename _Tp >`
`constexpr bool is_swappable_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_swappable_with_v`
- `template<typename _Tp >`
`constexpr bool is_unbounded_array_v`
- `constexpr memory_order memory_order_acq_rel`
- `constexpr memory_order memory_order_acquire`
- `constexpr memory_order memory_order_consume`
- `constexpr memory_order memory_order_relaxed`
- `constexpr memory_order memory_order_release`
- `constexpr memory_order memory_order_seq_cst`
- `constexpr nostopstate_t nostopstate`
- `const nothrow_t nothrow`
- `constexpr nullopt_t nullopt`
- `constexpr __cmp_cust:: _Partial_order partial_order`
- `constexpr piecewise_construct_t piecewise_construct`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_greater_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_greater_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_less_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_less_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_not_equal_v`
- `constexpr __cmp_cust:: _Strong_order strong_order`

- constexpr [try_to_lock_t](#) [try_to_lock](#)
- template<typename _Tp >
constexpr size_t **tuple_size_v**
- template<typename _Tp, size_t _Nm>
constexpr size_t **tuple_size_v**< [array](#)< _Tp, _Nm > >
- template<typename _Tp, size_t _Nm>
constexpr size_t **tuple_size_v**< [const array](#)< _Tp, _Nm > >
- template<typename _Tp1, typename _Tp2 >
constexpr size_t **tuple_size_v**< [const pair](#)< _Tp1, _Tp2 > >
- template<typename... _Types>
constexpr size_t **tuple_size_v**< [const tuple](#)< _Types... > >
- template<typename _Tp1, typename _Tp2 >
constexpr size_t **tuple_size_v**< [pair](#)< _Tp1, _Tp2 > >
- template<typename... _Types>
constexpr size_t **tuple_size_v**< [tuple](#)< _Types... > >
- constexpr [unreachable_sentinel_t](#) [unreachable_sentinel](#)
- constexpr size_t **variant_npos**
- template<typename _Variant >
constexpr size_t **variant_size_v**
- template<typename... _Types>
constexpr size_t **variant_size_v**< [const variant](#)< _Types... > >
- template<typename... _Types>
constexpr size_t **variant_size_v**< [variant](#)< _Types... > >
- constexpr [__cmp_cust::_Weak_order](#) **weak_order**

Standard Stream Objects

The `<iostream>` header declares the eight standard stream objects. For other declarations, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html> and the [I/O forward declarations](#)

They are required by default to cooperate with the global C library's `FILE` streams, and to be available during program startup and termination. For more information, see the section of the manual linked to above.

- [istream](#) [cin](#)
- [ostream](#) [cout](#)
- [ostream](#) [cerr](#)
- [ostream](#) [clog](#)
- [wistream](#) [wcin](#)
- [wostream](#) [wcout](#)
- [wostream](#) [wcerr](#)
- [wostream](#) [wclog](#)
- template<typename... _Bn>
constexpr bool **conjunction_v**
- template<typename... _Bn>
constexpr bool **disjunction_v**
- template<typename _Pp >
constexpr bool **negation_v**
- template<typename _Tp >
constexpr bool **is_void_v**
- template<typename _Tp >
constexpr bool **is_null_pointer_v**

- `template<typename _Tp >`
`constexpr bool is_integral_v`
- `template<typename _Tp >`
`constexpr bool is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool is_array_v`
- `template<typename _Tp >`
`constexpr bool is_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool is_rvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_enum_v`
- `template<typename _Tp >`
`constexpr bool is_union_v`
- `template<typename _Tp >`
`constexpr bool is_class_v`
- `template<typename _Tp >`
`constexpr bool is_function_v`
- `template<typename _Tp >`
`constexpr bool is_reference_v`
- `template<typename _Tp >`
`constexpr bool is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool is_object_v`
- `template<typename _Tp >`
`constexpr bool is_scalar_v`
- `template<typename _Tp >`
`constexpr bool is_compound_v`
- `template<typename _Tp >`
`constexpr bool is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_const_v`
- `template<typename _Tp >`
`constexpr bool is_volatile_v`
- `template<typename _Tp >`
`constexpr bool is_trivial_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool is_pod_v`
- `template<typename _Tp >`
`constexpr bool is_literal_type_v`

- `template<typename _Tp >`
`constexpr bool is_empty_v`
- `template<typename _Tp >`
`constexpr bool is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool is_abstract_v`
- `template<typename _Tp >`
`constexpr bool is_final_v`
- `template<typename _Tp >`
`constexpr bool is_signed_v`
- `template<typename _Tp >`
`constexpr bool is_unsigned_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_trivially_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_move_constructible_v`

- `template<typename _Tp, typename _Up >`
`constexpr bool is_nothrow_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr size_t alignment_of_v`
- `template<typename _Tp >`
`constexpr size_t rank_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t extent_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_same_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool is_base_of_v`
- `template<typename _From, typename _To >`
`constexpr bool is_convertible_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool is_invocable_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool is_nothrow_invocable_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool is_invocable_r_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool is_nothrow_invocable_r_v`

4.10.1 Detailed Description

ISO C++ entities toplevel namespace is `std`.

4.10.2 Typedef Documentation

`__ptr_rebind`

```
template<typename _Ptr, typename _Tp >
using std::__ptr_rebind = typedef typename pointer_traits<_Ptr>::template rebind<_Tp>
```

Convenience alias for rebinding pointers.

`__umap_traits`

```
template<bool _Cache>
using std::__umap_traits = typedef __detail::_Hashtable_traits<_Cache, false, true>
```

Base types for `unordered_map`.

`__ummap_traits`

```
template<bool _Cache>
using std::__ummap_traits = typedef __detail::_Hashtable_traits<_Cache, false, false>
```

Base types for `unordered_multimap`.

__umset_traits

```
template<bool _Cache>
using std::__umset_traits = typedef __detail::__Hashtable_traits<_Cache, true, false>
Base types for unordered_multiset.
```

__uset_traits

```
template<bool _Cache>
using std::__uset_traits = typedef __detail::__Hashtable_traits<_Cache, true, true>
Base types for unordered_set.
```

compare_three_way_result_t

```
template<typename _Tp , typename _Up = _Tp>
using std::compare_three_way_result_t = typedef typename __detail::__cmp3way_res_impl<_Tp, _Up>↵
::type
[cmp.result], result of three-way comparison
```

index_sequence

```
template<size_t... _Idx>
using std::index_sequence = typedef integer_sequence<size_t, _Idx...>
Alias template index_sequence.
```

index_sequence_for

```
template<typename... _Types>
using std::index_sequence_for = typedef make_index_sequence<sizeof...(_Types)>
Alias template index_sequence_for.
```

make_index_sequence

```
template<size_t _Num>
using std::make_index_sequence = typedef make_integer_sequence<size_t, _Num>
Alias template make_index_sequence.
```

make_integer_sequence

```
template<typename _Tp , _Tp _Num>
using std::make_integer_sequence = typedef __make_integer_seq<integer_sequence, _Tp, _Num>
Alias template make_integer_sequence.
```

new_handler

```
typedef void(* std::new_handler) ()
If you write your own error handler to be called by new, it must be of this type.
```

streamoff

```
typedef long long std::streamoff
Type used by fpos, char_traits<char>, and char_traits<wchar_t>.
In clauses 21.1.3.1 and 27.4.1 streamoff is described as an implementation defined type. Note: In versions of GCC up to and including GCC 3.3, streamoff was typedef long.
```

streampos

```
typedef fpos<mbstate_t> std::streampos
```

File position for char streams.

streamsize

```
typedef ptrdiff_t std::streamsize
```

Integral type for I/O operation counts and buffer sizes.

u16streampos

```
typedef fpos<mbstate_t> std::u16streampos
```

File position for char16_t streams.

u32streampos

```
typedef fpos<mbstate_t> std::u32streampos
```

File position for char32_t streams.

wstreampos

```
typedef fpos<mbstate_t> std::wstreampos
```

File position for wchar_t streams.

4.10.3 Enumeration Type Documentation**anonymous enum**

```
anonymous enum
```

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_↵_style.html This controls some aspect of the sort routines.

byte

```
enum class std::byte : unsigned char [strong]
```

std::byte

chars_format

```
enum class std::chars_format [strong]
```

floating-point format for primitive numerical conversion

float_denorm_style

```
enum std::float_denorm_style
```

Describes the denormalization for floating-point types.

These values represent the presence or absence of a variable number of exponent bits. This type is used in the std::numeric_limits class.

Enumerator

denorm_indeterminate	Indeterminate at compile time whether denormalized values are allowed.
denorm_absent	The type does not allow denormalized values.
denorm_present	The type allows denormalized values.

float_round_style

enum `std::float_round_style`

Describes the rounding style for floating-point types.

This is used in the `std::numeric_limits` class.

Enumerator

<code>round_toward_zero</code>	Intermediate.
<code>round_to_nearest</code>	To zero.
<code>round_toward_infinity</code>	To the nearest representable value.
<code>round_toward_neg_infinity</code>	To infinity.

io_errc

enum class `std::io_errc` [strong]

I/O error code.

4.10.4 Function Documentation**`__final_insertion_sort()`**

```
template<typename _RandomAccessIterator, typename _Compare>
constexpr void std::__final_insertion_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp) [constexpr]
```

This is a helper function for the sort routine.

References [__insertion_sort\(\)](#), and [__unguarded_insertion_sort\(\)](#).

`__find_if()` [1/2]

```
template<typename _InputIterator, typename _Predicate>
constexpr _InputIterator std::__find_if (
    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred,
    input_iterator_tag) [inline], [constexpr]
```

This is an overload used by find algos for the Input Iterator case.

Referenced by [__find_if_not\(\)](#), [__search_n_aux\(\)](#), [find\(\)](#), and [find_if\(\)](#).

`__find_if()` [2/2]

```
template<typename _RandomAccessIterator, typename _Predicate>
constexpr _RandomAccessIterator std::__find_if (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Predicate __pred,
    random_access_iterator_tag) [constexpr]
```

This is an overload used by find algos for the RAI case.

`__find_if_not()`

```
template<typename _InputIterator, typename _Predicate>
constexpr _InputIterator std::__find_if_not (
```



```

    _InputIterator __first,
    _InputIterator __last,
    _Predicate __pred ) [inline], [constexpr]

```

Provided for `stable_partition` to use.

References [__find_if\(\)](#), and [__iterator_category\(\)](#).

Referenced by [find_if_not\(\)](#).

`__find_if_not_n()`

```

template<typename _InputIterator , typename _Predicate , typename _Distance >
constexpr _InputIterator std::__find_if_not_n (
    _InputIterator __first,
    _Distance & __len,
    _Predicate __pred ) [constexpr]

```

Like `find_if_not()`, but uses and updates a count of the remaining range length instead of comparing against an end iterator.

Referenced by [__stable_partition_adaptive\(\)](#).

`__gcd()`

```

template<typename _EuclideanRingElement >
constexpr _EuclideanRingElement std::__gcd (
    _EuclideanRingElement __m,
    _EuclideanRingElement __n ) [constexpr]

```

This is a helper function for the rotate algorithm specialized on RAIs. It returns the greatest common divisor of two integer values.

`__gen_two_uniform_ints()`

```

template<typename _IntType , typename _UniformRandomBitGenerator >
pair< _IntType, _IntType > std::__gen_two_uniform_ints (
    _IntType __b0,
    _IntType __b1,
    _UniformRandomBitGenerator && __g )

```

Generate two uniformly distributed integers using a single distribution invocation.

Parameters

<code>__b0</code>	The upper bound for the first integer.
<code>__b1</code>	The upper bound for the second integer.
<code>__g</code>	A <code>UniformRandomBitGenerator</code> .

Returns

A pair (i, j) with i and j uniformly distributed over [0, __b0) and [0, __b1), respectively.

Requires: `__b0 * __b1 <= __g.max() - __g.min()`.

Using `uniform_int_distribution` with a range that is very small relative to the range of the generator ends up wasting potentially expensively generated randomness, since `uniform_int_distribution` does not store leftover randomness between invocations.

If we know we want two integers in ranges that are sufficiently small, we can compose the ranges, use a single distribution invocation, and significantly reduce the waste.

Referenced by [__sample\(\)](#), and [shuffle\(\)](#).

__heap_select()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::__heap_select (
    _RandomAccessIterator __first,
    _RandomAccessIterator __middle,
    _RandomAccessIterator __last,
    _Compare __comp ) [constexpr]
```

This is a helper function for the sort routines.

__inplace_stable_sort()

```
template<typename _RandomAccessIterator , typename _Compare >
void std::__inplace_stable_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp )
```

This is a helper function for the stable sorting routines.

References [__inplace_stable_sort\(\)](#), [__insertion_sort\(\)](#), and [__merge_without_buffer\(\)](#).

Referenced by [__inplace_stable_sort\(\)](#).

__insertion_sort()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::__insertion_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [constexpr]
```

This is a helper function for the sort routine.

References [__unguarded_linear_insert\(\)](#).

Referenced by [__final_insertion_sort\(\)](#), and [__inplace_stable_sort\(\)](#).

__introsort_loop()

```
template<typename _RandomAccessIterator , typename _Size , typename _Compare >
constexpr void std::__introsort_loop (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Size __depth_limit,
    _Compare __comp ) [constexpr]
```

This is a helper function for the sort routine.

References [__introsort_loop\(\)](#), and [__unguarded_partition_pivot\(\)](#).

Referenced by [__introsort_loop\(\)](#).

__lg()

```
constexpr int std::__lg (
    int __n ) [inline], [constexpr]
```

This is a helper function for the sort routines and for random.tcc.

Referenced by [nth_element\(\)](#), [nth_element\(\)](#), [std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >::operator\(\)\(\)](#), and [std::linear_congruential_engine<_UIntType, __a, __c, __m >::seed\(\)](#).

__merge_adaptive()

```
template<typename _BidirectionalIterator , typename _Distance , typename _Pointer , typename _↔
Compare >
```

```

void std::__merge_adaptive (
    _BidirectionalIterator __first,
    _BidirectionalIterator __middle,
    _BidirectionalIterator __last,
    _Distance __len1,
    _Distance __len2,
    _Pointer __buffer,
    _Distance __buffer_size,
    _Compare __comp )

```

This is a helper function for the merge routines.

References [__merge_adaptive\(\)](#), [__move_merge_adaptive\(\)](#), [__move_merge_adaptive_backward\(\)](#), [__rotate_adaptive\(\)](#), [advance\(\)](#), and [distance\(\)](#).

Referenced by [__merge_adaptive\(\)](#).

[__merge_without_buffer\(\)](#)

```

template<typename _BidirectionalIterator , typename _Distance , typename _Compare >
void std::__merge_without_buffer (
    _BidirectionalIterator __first,
    _BidirectionalIterator __middle,
    _BidirectionalIterator __last,
    _Distance __len1,
    _Distance __len2,
    _Compare __comp )

```

This is a helper function for the merge routines.

References [__merge_without_buffer\(\)](#), [advance\(\)](#), and [distance\(\)](#).

Referenced by [__inplace_stable_sort\(\)](#), and [__merge_without_buffer\(\)](#).

[__move_median_to_first\(\)](#)

```

template<typename _Iterator , typename _Compare >
constexpr void std::__move_median_to_first (
    _Iterator __result,
    _Iterator __a,
    _Iterator __b,
    _Iterator __c,
    _Compare __comp ) [constexpr]

```

Swaps the median value of [*__a](#), [*__b](#) and [*__c](#) under [__comp](#) to [*__result](#).

Referenced by [__unguarded_partition_pivot\(\)](#).

[__move_merge\(\)](#)

```

template<typename _InputIterator , typename _OutputIterator , typename _Compare >
_OutputIterator std::__move_merge (
    _InputIterator __first1,
    _InputIterator __last1,
    _InputIterator __first2,
    _InputIterator __last2,
    _OutputIterator __result,
    _Compare __comp )

```

This is a helper function for the [__merge_sort_loop](#) routines.

[__move_merge_adaptive\(\)](#)

```

template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename

```

```

_Compare >
void std::__move_merge_adaptive (
    _InputIterator1 __first1,
    _InputIterator1 __last1,
    _InputIterator2 __first2,
    _InputIterator2 __last2,
    _OutputIterator __result,
    _Compare __comp )

```

This is a helper function for the `__merge_adaptive` routines.
Referenced by [__merge_adaptive\(\)](#).

`__move_merge_adaptive_backward()`

```

template<typename _BidirectionalIterator1 , typename _BidirectionalIterator2 , typename _BidirectionalIterator3 , typename _Compare >
void std::__move_merge_adaptive_backward (
    _BidirectionalIterator1 __first1,
    _BidirectionalIterator1 __last1,
    _BidirectionalIterator2 __first2,
    _BidirectionalIterator2 __last2,
    _BidirectionalIterator3 __result,
    _Compare __comp )

```

This is a helper function for the `__merge_adaptive` routines.
Referenced by [__merge_adaptive\(\)](#).

`__partition()` [1/2]

```

template<typename _BidirectionalIterator , typename _Predicate >
constexpr _BidirectionalIterator std::__partition (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last,
    _Predicate __pred,
    bidirectional\_iterator\_tag ) [constexpr]

```

This is a helper function...

`__partition()` [2/2]

```

template<typename _ForwardIterator , typename _Predicate >
constexpr _ForwardIterator std::__partition (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred,
    forward\_iterator\_tag ) [constexpr]

```

This is a helper function...

Referenced by [partition\(\)](#).

`__reverse()` [1/2]

```

template<typename _BidirectionalIterator >
constexpr void std::__reverse (
    _BidirectionalIterator __first,
    _BidirectionalIterator __last,
    bidirectional\_iterator\_tag ) [constexpr]

```

This is an uglified `reverse(_BidirectionalIterator, _BidirectionalIterator)` overloaded for bidirectional iterators.
Referenced by [__rotate\(\)](#), and [reverse\(\)](#).

__reverse() [2/2]

```
template<typename _RandomAccessIterator >
constexpr void std::__reverse (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    random_access_iterator_tag ) [constexpr]
```

This is an uglified reverse([_BidirectionalIterator](#), [_BidirectionalIterator](#)) overloaded for random access iterators.

__rotate() [1/3]

```
template<typename _BidirectionalIterator >
constexpr _BidirectionalIterator std::_V2::__rotate (
    _BidirectionalIterator __first,
    _BidirectionalIterator __middle,
    _BidirectionalIterator __last,
    bidirectional_iterator_tag ) [constexpr]
```

This is a helper function for the rotate algorithm.

References [__reverse\(\)](#), and [__rotate\(\)](#).

__rotate() [2/3]

```
template<typename _ForwardIterator >
constexpr _ForwardIterator std::_V2::__rotate (
    _ForwardIterator __first,
    _ForwardIterator __middle,
    _ForwardIterator __last,
    forward_iterator_tag ) [constexpr]
```

This is a helper function for the rotate algorithm.

References [__rotate\(\)](#).

Referenced by [__rotate\(\)](#), [__rotate\(\)](#), [__rotate\(\)](#), and [rotate\(\)](#).

__rotate() [3/3]

```
template<typename _RandomAccessIterator >
constexpr _RandomAccessIterator std::_V2::__rotate (
    _RandomAccessIterator __first,
    _RandomAccessIterator __middle,
    _RandomAccessIterator __last,
    random_access_iterator_tag ) [constexpr]
```

This is a helper function for the rotate algorithm.

References [__rotate\(\)](#), and [swap\(\)](#).

__rotate_adaptive()

```
template<typename _BidirectionalIterator1 , typename _BidirectionalIterator2 , typename _Distance >
_BidirectionalIterator1 std::__rotate_adaptive (
    _BidirectionalIterator1 __first,
    _BidirectionalIterator1 __middle,
    _BidirectionalIterator1 __last,
    _Distance __len1,
    _Distance __len2,
    _BidirectionalIterator2 __buffer,
    _Distance __buffer_size )
```

This is a helper function for the merge routines.

Referenced by [__merge_adaptive\(\)](#).

__sample() [1/2]

```
template<typename _ForwardIterator , typename _OutputIterator , typename _Cat , typename _Size ,
typename _UniformRandomBitGenerator >
_OutputIterator std::__sample (
    _ForwardIterator __first,
    _ForwardIterator __last,
    forward_iterator_tag ,
    _OutputIterator __out,
    _Cat ,
    _Size __n,
    _UniformRandomBitGenerator && __g )
```

Selection sampling algorithm.

References [__gen_two_uniform_ints\(\)](#), [distance\(\)](#), [std::pair<_T1, _T2>::first](#), [min\(\)](#), and [std::pair<_T1, _T2>::second](#).

__sample() [2/2]

```
template<typename _InputIterator , typename _RandomAccessIterator , typename _Size , typename _↵
UniformRandomBitGenerator >
_RandomAccessIterator std::__sample (
    _InputIterator __first,
    _InputIterator __last,
    input_iterator_tag ,
    _RandomAccessIterator __out,
    random_access_iterator_tag ,
    _Size __n,
    _UniformRandomBitGenerator && __g )
```

Reservoir sampling algorithm.

__search_n_aux() [1/2]

```
template<typename _ForwardIterator , typename _Integer , typename _UnaryPredicate >
constexpr _ForwardIterator std::__search_n_aux (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Integer __count,
    _UnaryPredicate __unary_pred,
    std::forward_iterator_tag ) [constexpr]
```

This is an helper function for `search_n` overloaded for forward iterators.

References [__find_if\(\)](#).

__search_n_aux() [2/2]

```
template<typename _RandomAccessIter , typename _Integer , typename _UnaryPredicate >
constexpr _RandomAccessIter std::__search_n_aux (
    _RandomAccessIter __first,
    _RandomAccessIter __last,
    _Integer __count,
    _UnaryPredicate __unary_pred,
    std::random_access_iterator_tag ) [constexpr]
```

This is an helper function for `search_n` overloaded for random access iterators.

__stable_partition_adaptive()

```
template<typename _ForwardIterator , typename _Pointer , typename _Predicate , typename _Distance
>
_FForwardIterator std::__stable_partition_adaptive (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Predicate __pred,
    _Distance __len,
    _Pointer __buffer,
    _Distance __buffer_size )
```

This is a helper function... Requires `__first != __last` and `!__pred(__first)` and `__len == distance(__first, __last)`.

`!__pred(__first)` allows us to guarantee that we don't move-assign an element onto itself.

References [__find_if_not_n\(\)](#), [__stable_partition_adaptive\(\)](#), and [advance\(\)](#).

Referenced by [__stable_partition_adaptive\(\)](#).

__unguarded_insertion_sort()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::__unguarded_insertion_sort (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

This is a helper function for the sort routine.

References [__unguarded_linear_insert\(\)](#).

Referenced by [__final_insertion_sort\(\)](#).

__unguarded_linear_insert()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr void std::__unguarded_linear_insert (
    _RandomAccessIterator __last,
    _Compare __comp ) [constexpr]
```

This is a helper function for the sort routine.

Referenced by [__insertion_sort\(\)](#), and [__unguarded_insertion_sort\(\)](#).

__unguarded_partition()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr _RandomAccessIterator std::__unguarded_partition (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _RandomAccessIterator __pivot,
    _Compare __comp ) [constexpr]
```

This is a helper function...

Referenced by [__unguarded_partition_pivot\(\)](#).

__unguarded_partition_pivot()

```
template<typename _RandomAccessIterator , typename _Compare >
constexpr _RandomAccessIterator std::__unguarded_partition_pivot (
    _RandomAccessIterator __first,
    _RandomAccessIterator __last,
    _Compare __comp ) [inline], [constexpr]
```

This is a helper function...

References [__move_median_to_first\(\)](#), and [__unguarded_partition\(\)](#).
 Referenced by [__introsort_loop\(\)](#).

`__unique_copy()` [1/3]

```
template<typename _ForwardIterator , typename _OutputIterator , typename _BinaryPredicate >
constexpr _OutputIterator std::__unique_copy (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _OutputIterator __result,
    _BinaryPredicate __binary_pred,
    forward_iterator_tag ,
    output_iterator_tag ) [constexpr]
```

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for forward iterators and output iterator as result.

Referenced by [unique_copy\(\)](#), and [unique_copy\(\)](#).

`__unique_copy()` [2/3]

```
template<typename _InputIterator , typename _ForwardIterator , typename _BinaryPredicate >
constexpr _ForwardIterator std::__unique_copy (
    _InputIterator __first,
    _InputIterator __last,
    _ForwardIterator __result,
    _BinaryPredicate __binary_pred,
    input_iterator_tag ,
    forward_iterator_tag ) [constexpr]
```

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for input iterators and forward iterator as result.

`__unique_copy()` [3/3]

```
template<typename _InputIterator , typename _OutputIterator , typename _BinaryPredicate >
constexpr _OutputIterator std::__unique_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    _BinaryPredicate __binary_pred,
    input_iterator_tag ,
    output_iterator_tag ) [constexpr]
```

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for input iterators and output iterator as result.

`_Construct()`

```
template<typename _Tp , typename... _Args>
constexpr void std::_Construct (
    _Tp * __p,
    _Args &&... __args ) [inline], [constexpr]
```

Constructs an object in existing memory by invoking an allocated object's constructor with an initializer.

Referenced by [std::allocator_traits< allocator< void > >::construct\(\)](#).

`_Destroy()` [1/3]

```
template<typename _ForwardIterator >
```



```
constexpr void std::_Destroy (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr]
```

Destroy a range of objects. If the value_type of the object has a trivial destructor, the compiler should optimize all of this away, otherwise the objects' destructors must be invoked.

`_Destroy()` [2/3]

```
template<typename _ForwardIterator , typename _Allocator >
constexpr void std::_Destroy (
    _ForwardIterator __first,
    _ForwardIterator __last,
    _Allocator & __alloc ) [constexpr]
```

Destroy a range of objects using the supplied allocator. For non-default allocators we do not optimize away invocation of `destroy()` even if `_Tp` has a trivial destructor.

References `__addressof()`, and `std::allocator_traits<_Alloc>::destroy()`.

Referenced by `std::vector<_Tp, _Alloc>::~~vector()`, `std::deque<_Tp, _Alloc>::_M_fill_initialize()`, `std::deque<_Tp, _Alloc>::_M_range_initialize()`, `std::allocator_traits<allocator<void>>::destroy()`, `std::vector<_State<_TraitsT::char_type>>::operator=()`, `std::vector<_Tp, _Alloc>::operator=()`, and `std::vector<_Tp, _Alloc>::reserve()`.

`_Destroy()` [3/3]

```
template<typename _Tp >
constexpr void std::_Destroy (
    _Tp * __pointer ) [inline], [constexpr]
```

Destroy the object pointed to by a pointer type.

`_Destroy_n()`

```
template<typename _ForwardIterator , typename _Size >
constexpr _ForwardIterator std::_Destroy_n (
    _ForwardIterator __first,
    _Size __count ) [inline], [constexpr]
```

Destroy a range of objects. If the value_type of the object has a trivial destructor, the compiler should optimize all of this away, otherwise the objects' destructors must be invoked.

`acos()`

```
template<typename _Tp >
std::complex<_Tp> std::acos (
    const std::complex<_Tp> & __z ) [inline]
acos(__z) [8.1.2].
```

`acosh()`

```
template<typename _Tp >
std::complex<_Tp> std::acosh (
    const std::complex<_Tp> & __z ) [inline]
acosh(__z) [8.1.5].
```

`advance()`

```
template<typename _InputIterator , typename _Distance >
constexpr void std::advance (
```

```

    _InputIterator & __i,
    _Distance __n ) [inline], [constexpr]

```

A generalization of pointer arithmetic.

Parameters

\leftrightarrow _i	An input iterator.
\leftrightarrow _n	The <i>delta</i> by which to change __i.

Returns

Nothing.

This increments *i* by *n*. For bidirectional and random access iterators, __n may be negative, in which case __i is decremented.

For random access iterators, this uses their + and – operations and are constant time. For other iterator classes they are linear time.

References [_iterator_category\(\)](#).

Referenced by [__merge_adaptive\(\)](#), [__merge_without_buffer\(\)](#), [__stable_partition_adaptive\(\)](#), [std::deque<_Tp, _Alloc>::_M_range_init\(\)](#), [__gnu_pbds::detail::pat_trie_base::_Node_citer<Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc>::get_child\(\)](#), [__gnu_pbds::detail::pat_trie_base::_Node_iter<Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc>::get_child\(\)](#), [std::vector<_State<_TraitsT::char_type>>::operator=\(\)](#), and [partition_point\(\)](#).

arg()

```

template<typename _Tp >
__gnu_cxx::__promote<_Tp>::__type std::arg (
    _Tp __x ) [inline]

```

Additional overloads [8.1.9].

asin()

```

template<typename _Tp >
std::complex<_Tp> std::asin (
    const std::complex<_Tp> & __z ) [inline]
asin(__z) [8.1.3].

```

asinh()

```

template<typename _Tp >
std::complex<_Tp> std::asinh (
    const std::complex<_Tp> & __z ) [inline]
asinh(__z) [8.1.6].

```

atan()

```

template<typename _Tp >
std::complex<_Tp> std::atan (
    const std::complex<_Tp> & __z ) [inline]
atan(__z) [8.1.4].

```

atanh()

```
template<typename _Tp >
std::complex< _Tp > std::atanh (
    const std::complex< _Tp > & __z ) [inline]
atanh(__z) [8.1.7].
```

begin() [1/3]

```
template<typename _Container >
constexpr auto std::begin (
    _Container & __cont ) -> decltype(__cont.begin()) [inline], [constexpr]
```

Return an iterator pointing to the first element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

begin() [2/3]

```
template<typename _Tp , size_t _Nm>
constexpr _Tp * std::begin (
    _Tp(&) __arr[_Nm] ) [inline], [constexpr], [noexcept]
```

Return an iterator pointing to the first element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

begin() [3/3]

```
template<typename _Container >
constexpr auto std::begin (
    const _Container & __cont ) -> decltype(__cont.begin()) [inline], [constexpr]
```

Return an iterator pointing to the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

bind_front()

```
template<typename _Fn , typename... _Args>
constexpr _Bind_front_t< _Fn, _Args... > std::bind_front (
    _Fn && __fn,
    _Args &&... __args ) [constexpr], [noexcept]
```

Create call wrapper by partial application of arguments to function.

The result of `std::bind_front(f, args...)` is a function object that stores `f` and the bound arguments, `args...`. When that function object is invoked with `call_args...` it returns the result of calling `f(args..., call_args...)`.

Since

C++20

boolalpha()

```
ios_base & std::boolalpha (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::boolalpha)`.

References [std::ios_base::boolalpha](#).

cbegin()

```
template<typename _Container >
constexpr auto std::cbegin (
    const _Container & __cont ) -> decltype(std::begin(__cont)) [constexpr], [noexcept]
```

Return an iterator pointing to the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

References [begin\(\)](#).

Referenced by [std::vector<_Tp, _Alloc>::insert\(\)](#).

cend()

```
template<typename _Container >
constexpr auto std::cend (
    const _Container & __cont ) -> decltype(std::end(__cont)) [constexpr], [noexcept]
```

Return an iterator pointing to one past the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

References [end\(\)](#).

const_pointer_cast()

```
template<typename _Tp , typename _Tp1 , _Lock_policy _Lp>
__shared_ptr< _Tp, _Lp > std::const_pointer_cast (
    const __shared_ptr< _Tp1, _Lp > & __r ) [inline], [noexcept]
const_pointer_cast
```

crbegin()

```
template<typename _Container >
constexpr auto std::crbegin (
    const _Container & __cont ) -> decltype(std::rbegin(__cont)) [inline], [constexpr]
```

Return a reverse iterator pointing to the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

References [rbegin\(\)](#).

crend()

```
template<typename _Container >
constexpr auto std::crend (
    const _Container & __cont ) -> decltype(std::rend(__cont))    [inline], [constexpr]
```

Return a reverse iterator pointing one past the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

References [rend\(\)](#).

data() [1/4]

```
template<typename _Container >
constexpr auto std::data (
    _Container & __cont ) -> decltype(__cont.data())    [constexpr], [noexcept]
```

Return the data pointer of a container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#).

data() [2/4]

```
template<typename _Tp , size_t _Nm>
constexpr _Tp * std::data (
    _Tp(&) __array[_Nm] )    [constexpr], [noexcept]
```

Return the data pointer of an array.

Parameters

<code>__array</code>	Array.
----------------------	--------

data() [3/4]

```
template<typename _Container >
constexpr auto std::data (
    const _Container & __cont ) -> decltype(__cont.data())    [constexpr], [noexcept]
```

Return the data pointer of a const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

data() [4/4]

```
template<typename _Tp >
constexpr const _Tp * std::data (
    initializer_list< _Tp > __il ) [constexpr], [noexcept]
```

Return the data pointer of an initializer list.

Parameters

<code>__il</code>	Initializer list.
-------------------	-------------------

dec()

```
ios_base & std::dec (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::dec, ios_base::basefield)`.

References [std::ios_base::basefield](#), and [std::ios_base::dec](#).

defaultfloat()

```
ios_base & std::defaultfloat (
    ios_base & __base ) [inline]
```

Calls `base.unsetf(ios_base::floatfield)`.

References [std::ios_base::floatfield](#).

distance()

```
template<typename _InputIterator >
constexpr iterator_traits< _InputIterator >::difference_type std::distance (
    _InputIterator __first,
    _InputIterator __last ) [inline], [constexpr]
```

A generalization of pointer arithmetic.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

The distance between them.

Returns `n` such that `__first + n == __last`. This requires that `__last` must be reachable from `__first`. Note that `n` may be negative.

For random access iterators, this uses their `+` and `-` operations and are constant time. For other iterator classes they are linear time.

References [__iterator_category\(\)](#).

Referenced by [__merge_adaptive\(\)](#), [__merge_without_buffer\(\)](#), [__sample\(\)](#), [std::deque< _Tp, _Alloc >::M_range_initialize\(\)](#), [is_heap\(\)](#), [is_heap_until\(\)](#), [is_heap_until\(\)](#), [std::sub_match< _Biter >::length\(\)](#), [__gnu_parallel::multiseq_partition\(\)](#),

[__gnu_parallel::multiseq_selection\(\)](#), [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::operator=\(\)](#), [partition_point\(\)](#), and [std::match_results< _Bi_iter, _Alloc >::position\(\)](#).

dynamic_pointer_cast()

```
template<typename _Tp , typename _Tp1 , _Lock_policy _Lp>
__shared_ptr< _Tp, _Lp > std::dynamic_pointer_cast (
    const __shared_ptr< _Tp1, _Lp > & __r ) [inline], [noexcept]
dynamic_pointer_cast
```

empty() [1/3]

```
template<typename _Container >
constexpr auto std::empty (
    const _Container & __cont ) -> decltype(__cont.empty()) [constexpr], [noexcept]
```

Return whether a container is empty.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Referenced by [std::list< _Tp, _Alloc >::sort\(\)](#), and [std::list< _Tp, _Alloc >::sort\(\)](#).

empty() [2/3]

```
template<typename _Tp , size_t _Nm>
constexpr bool std::empty (
    const _Tp (&) [_Nm] ) [constexpr], [noexcept]
```

Return whether an array is empty (always false).

empty() [3/3]

```
template<typename _Tp >
constexpr bool std::empty (
    initializer_list< _Tp > __il ) [constexpr], [noexcept]
```

Return whether an initializer_list is empty.

Parameters

<code>__il</code>	Initializer list.
-------------------	-------------------

end() [1/3]

```
template<typename _Container >
constexpr auto std::end (
    _Container & __cont ) -> decltype(__cont.end()) [inline], [constexpr]
```

Return an iterator pointing to one past the last element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

end() [2/3]

```
template<typename _Tp , size_t _Nm>
```

```
constexpr _Tp * std::end (
    _Tp(&) __arr[_Nm] ) [inline], [constexpr], [noexcept]
```

Return an iterator pointing to one past the last element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

end() [3/3]

```
template<typename _Container >
constexpr auto std::end (
    const _Container & __cont ) -> decltype(__cont.end()) [inline], [constexpr]
```

Return an iterator pointing to one past the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

endl()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::endl (
    basic_ostream< _CharT, _Traits > & __os ) [inline]
```

Write a newline and flush the stream.

This manipulator is often mistakenly used when a simple newline is desired, leading to poor buffering performance. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.↵streambuf.buffering> for more on this subject.

ends()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::ends (
    basic_ostream< _CharT, _Traits > & __os ) [inline]
```

Write a null character into the output sequence.

Null character is `CharT()` by definition. For `CharT` of `char`, this correctly writes the ASCII NUL character string terminator.

exchange()

```
template<typename _Tp , typename _Up = _Tp>
constexpr _Tp std::exchange (
    _Tp & __obj,
    _Up && __new_val ) [inline], [constexpr], [noexcept]
```

Assign `__new_val` to `__obj` and return its previous value.

fabs()

```
template<typename _Tp >
_Tp std::fabs (
    const std::complex< _Tp > & __z ) [inline]
fabs(__z) [8.1.8].
```


fixed()

```
ios_base & std::fixed (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::fixed, ios_base::floatfield)`.

References `std::ios_base::fixed`, `fixed()`, and `std::ios_base::floatfield`.

Referenced by `fixed()`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::flush (
    basic_ostream< _CharT, _Traits > & __os ) [inline]
```

Flushes the output stream.

This manipulator simply calls the stream's `flush()` member function.

from_chars()

```
template<typename _Tp >
__detail::__integer_from_chars_result_type< _Tp > std::from_chars (
    const char * __first,
    const char * __last,
    _Tp & __value,
    int __base = 10 )
```

`std::from_chars` for integral types.

gcd()

```
template<typename _Mn , typename _Nn >
constexpr common_type_t< _Mn, _Nn > std::gcd (
    _Mn __m,
    _Nn __n ) [constexpr], [noexcept]
```

Greatest common divisor.

get() [1/12]

```
template<typename _Tp , typename _Up >
constexpr const _Tp && std::get (
    const pair< _Tp, _Up > && __p ) [constexpr], [noexcept]
```

`std::get` overloads for accessing members of `std::pair`

References `move()`.

get() [2/12]

```
template<typename _Tp , typename _Up >
constexpr const _Tp & std::get (
    const pair< _Tp, _Up > & __p ) [constexpr], [noexcept]
```

`std::get` overloads for accessing members of `std::pair`

get() [3/12]

```
template<size_t _Int, class _Tp1 , class _Tp2 >
constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && std::get (
    const pair< _Tp1, _Tp2 > && __in ) [constexpr], [noexcept]
```

`std::get` overloads for accessing members of `std::pair`

References `move()`.

get() [4/12]

```
template<size_t _Int, class _Tp1, class _Tp2 >
constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & std::get (
    const pair< _Tp1, _Tp2 > & __in ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
```

get() [5/12]

```
template<typename _Tp, typename _Up >
constexpr const _Tp && std::get (
    const pair< _Up, _Tp > && __p ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
References move\(\).
```

get() [6/12]

```
template<typename _Tp, typename _Up >
constexpr const _Tp & std::get (
    const pair< _Up, _Tp > & __p ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
```

get() [7/12]

```
template<typename _Tp, typename _Up >
constexpr _Tp && std::get (
    pair< _Tp, _Up > && __p ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
References move\(\).
```

get() [8/12]

```
template<typename _Tp, typename _Up >
constexpr _Tp & std::get (
    pair< _Tp, _Up > & __p ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
```

get() [9/12]

```
template<size_t _Int, class _Tp1, class _Tp2 >
constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && std::get (
    pair< _Tp1, _Tp2 > && __in ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
References move\(\).
```

get() [10/12]

```
template<size_t _Int, class _Tp1, class _Tp2 >
constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & std::get (
    pair< _Tp1, _Tp2 > & __in ) [constexpr], [noexcept]
std::get overloads for accessing members of std::pair
```

get() [11/12]

```
template<typename _Tp , typename _Up >
constexpr _Tp && std::get (
    pair< _Up, _Tp > && __p ) [constexpr], [noexcept]
```

std::get overloads for accessing members of std::pair

References [move\(\)](#).

get() [12/12]

```
template<typename _Tp , typename _Up >
constexpr _Tp & std::get (
    pair< _Up, _Tp > & __p ) [constexpr], [noexcept]
```

std::get overloads for accessing members of std::pair

get_money()

```
template<typename _MoneyT >
_Get_money< _MoneyT > std::get_money (
    _MoneyT & __mon,
    bool __intl = false ) [inline]
```

Extended manipulator for extracting money.

Parameters

<code>__mon</code>	Either long double or a specialization of <code>basic_string</code> .
<code>__intl</code>	A bool indicating whether international format is to be used.

Sent to a stream object, this manipulator extracts `__mon`.

get_new_handler()

```
new_handler std::get_new_handler ( ) [noexcept]
```

Return the current new handler.

get_temporary_buffer()

```
template<typename _Tp >
pair< _Tp *, ptrdiff_t > std::get_temporary_buffer (
    ptrdiff_t __len ) [noexcept]
```

Allocates a temporary buffer.

Parameters

<code>__len</code>	The number of objects of type <code>Tp</code> .
--------------------	---

Returns

See full description.

Reinventing the wheel, but this time with prettier spokes!

This function tries to obtain storage for `__len` adjacent `Tp` objects. The objects themselves are not constructed, of course. A `pair<>` is returned containing *the buffer's address and capacity (in the units of `sizeof(_Tp)`)*, or a pair of 0 values if no storage can be obtained. Note that the capacity obtained may be less than that requested if the memory is unavailable; you should compare `len` with the `.second` return value.

Provides the nothrow exception guarantee.

get_time()

```
template<typename _CharT >
_Get_time< _CharT > std::get_time (
    std::tm * __tmb,
    const _CharT * __fmt ) [inline]
```

Extended manipulator for extracting time.

This manipulator uses `time_get::get` to extract time. [ext.manip]

Parameters

<code>__tmb</code>	struct to extract the time data to.
<code>__fmt</code>	format string.

getline() [1/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (
    basic_istream< _CharT, _Traits > && __is,
    basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Read a line from an rvalue stream into a string.

References [getline\(\)](#).

getline() [2/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (
    basic_istream< _CharT, _Traits > && __is,
    basic_string< _CharT, _Traits, _Alloc > & __str,
    _CharT __delim ) [inline]
```

Read a line from an rvalue stream into a string.

References [getline\(\)](#).

getline() [3/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
basic_istream< _CharT, _Traits > & std::getline (
    basic_istream< _CharT, _Traits > & __is,
    __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `is` into `__str` until '

' is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased. If end of line was encountered, it is extracted but not stored into `__str`.

References [getline\(\)](#), and [std::basic_ios<_CharT, _Traits>::widen\(\)](#).

getline() [4/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
basic_istream< _CharT, _Traits > & std::getline (
    basic_istream< _CharT, _Traits > & __is,
    __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    _CharT __delim )
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.
<code>__delim</code>	Character marking end of line.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until `__delim` is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased. If `delim` was encountered, it is extracted but not stored into `__str`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::max_size\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

getline() [5/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (
    basic_istream< _CharT, _Traits > & __is,
    basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `is` into `__str` until '

' is found, the end of the stream is encountered, or `str.max_size()` is reached. Any previous contents of `__str` are erased.

If end of line is encountered, it is extracted but not stored into `__str`.

References [getline\(\)](#), and [std::basic_ios<_CharT, _Traits>::widen\(\)](#).

getline() [6/6]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::getline (
    basic_istream<_CharT, _Traits> & __is,
    basic_string<_CharT, _Traits, _Alloc> & __str,
    _CharT __delim )
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.
<code>__delim</code>	Character marking end of line.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until `__delim` is found, the end of the stream is encountered, or `str.max_size()` is reached. Any previous contents of `__str` are erased. If `__delim` is encountered, it is extracted but not stored into `__str`.

References [std::basic_string<_CharT, _Traits, _Alloc>::erase\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::max_size\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

Referenced by [getline\(\)](#), [getline\(\)](#), [getline\(\)](#), and [getline\(\)](#).

hex()

```
ios_base & std::hex (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::hex, ios_base::basefield)`.

References [std::ios_base::basefield](#), [std::ios_base::hex](#), and [hex\(\)](#).

Referenced by [hex\(\)](#).

hexfloat()

```
ios_base & std::hexfloat (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::fixed|ios_base::scientific, ios_base::floatfield)`

References [std::ios_base::fixed](#), [std::ios_base::floatfield](#), and [std::ios_base::scientific](#).

internal()

```
ios_base & std::internal (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::internal, ios_base::adjustfield)`.

References [std::ios_base::adjustfield](#), and [std::ios_base::internal](#).

invoke()

```
template<typename _Callable , typename... _Args>
constexpr invoke_result_t< _Callable, _Args... > std::invoke (
    _Callable && __fn,
    _Args &&... __args ) [inline], [constexpr], [noexcept]
```

Invoke a callable object.

`std::invoke` takes a callable object as its first argument and calls it with the remaining arguments. The callable object can be a pointer or reference to a function, a lambda closure, a class with `operator()`, or even a pointer-to-member. For a pointer-to-member the first argument must be a reference or pointer to the object that the pointer-to-member will be applied to.

Since

C++17

isalnum()

```
template<typename _CharT >
bool std::isalnum (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::alnum, __c)`.

isalpha()

```
template<typename _CharT >
bool std::isalpha (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::alpha, __c)`.

isblank()

```
template<typename _CharT >
bool std::isblank (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::blank, __c)`.

isctrl()

```
template<typename _CharT >
bool std::isctrl (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::cntrl, __c)`.

isdigit()

```
template<typename _CharT >
bool std::isdigit (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::digit, __c)`.

isgraph()

```
template<typename _CharT >
bool std::isgraph (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::graph, __c)`.

islower()

```
template<typename _CharT >
bool std::islower (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::lower, __c)`.

isprint()

```
template<typename _CharT >
bool std::isprint (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::print, __c)`.

ispunct()

```
template<typename _CharT >
bool std::ispunct (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::punct, __c)`.

isspace()

```
template<typename _CharT >
bool std::isspace (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::space, __c)`.

isupper()

```
template<typename _CharT >
bool std::isupper (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::upper, __c)`.

isxdigit()

```
template<typename _CharT >
bool std::isxdigit (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype.is(ctype_base::xdigit, __c)`.

launder()

```
template<typename _Tp >
constexpr _Tp * std::launder (
    _Tp * __p ) [constexpr], [noexcept]
```

Pointer optimization barrier [ptr.launder].

lcm()

```
template<typename _Mn , typename _Nn >
constexpr common_type_t< _Mn, _Nn > std::lcm (
    _Mn __m,
    _Nn __n ) [constexpr], [noexcept]
```

Least common multiple.

left()

```
ios_base & std::left (
    ios_base & __base ) [inline]
```

Calls base.setf(ios_base::left, ios_base::adjustfield).

References [std::ios_base::adjustfield](#), and [std::ios_base::left](#).

noboolalpha()

```
ios_base & std::noboolalpha (
    ios_base & __base ) [inline]
```

Calls base.unsetf(ios_base::boolalpha).

References [std::ios_base::boolalpha](#).

noshowbase()

```
ios_base & std::noshowbase (
    ios_base & __base ) [inline]
```

Calls base.unsetf(ios_base::showbase).

References [std::ios_base::showbase](#).

noshowpoint()

```
ios_base & std::noshowpoint (
    ios_base & __base ) [inline]
```

Calls base.unsetf(ios_base::showpoint).

References [std::ios_base::showpoint](#).

noshowpos()

```
ios_base & std::noshowpos (
    ios_base & __base ) [inline]
```

Calls base.unsetf(ios_base::showpos).

References [std::ios_base::showpos](#).

noskipws()

```
ios_base & std::noskipws (
    ios_base & __base ) [inline]
```

Calls base.unsetf(ios_base::skipws).

References [std::ios_base::skipws](#).

nounitbuf()

```
ios_base & std::nounitbuf (
    ios_base & __base ) [inline]
```

Calls `base.unsetf(ios_base::unitbuf)`.

References [std::ios_base::unitbuf](#).

nouppercase()

```
ios_base & std::nouppercase (
    ios_base & __base ) [inline]
```

Calls `base.unsetf(ios_base::uppercase)`.

References [std::ios_base::uppercase](#).

oct()

```
ios_base & std::oct (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::oct, ios_base::basefield)`.

References [std::ios_base::basefield](#), and [std::ios_base::oct](#).

operator"!="() [1/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator!= (
    const _CharT * __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test difference of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) != 0`. False otherwise.

operator"!="() [2/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator!= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test difference of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

operator"!="() [3/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator!= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [noexcept]
```

Test difference of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

operator"!="() [4/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator!= (
    const deque< _Tp, _Alloc > & __x,
    const deque< _Tp, _Alloc > & __y ) [inline]
```

Based on `operator==`.

operator"!="() [5/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator!= (
    const forward_list< _Tp, _Alloc > & __lx,
    const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Based on `operator==`.

operator"!="() [6/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator!= (
    const list< _Tp, _Alloc > & __x,
    const list< _Tp, _Alloc > & __y ) [inline]
```

Based on `operator==`.

operator"!="() [7/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator!= (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on `operator==`.

operator!=() [8/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator!= (
    const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator==.

operator!=() [9/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator!= (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(x == y).

operator!=() [10/13]

```
template<typename _Tp , typename _Seq >
bool std::operator!= (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Based on operator==.

operator!=() [11/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator!= (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(x == y).

operator!=() [12/13]

```
template<typename _Tp , typename _Seq >
bool std::operator!= (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Based on operator==.

operator!=() [13/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator!= (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline]
```

Based on operator==.

operator&()

```
template<size_t _Nb>
bitset< _Nb > std::operator& (
    const bitset< _Nb > & __x,
    const bitset< _Nb > & __y ) [inline], [noexcept]
```

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

Returns

A new bitset.

These should be self-explanatory.

operator+() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (
    _CharT __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [constexpr]
Concatenate character and string.
```

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

References `std::basic_string< _CharT, _Traits, _Alloc >::get_allocator()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

operator+() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (
    const _CharT * __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [constexpr]
Concatenate C string and string.
```

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

References `std::basic_string< _CharT, _Traits, _Alloc >::get_allocator()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

operator+() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (
```

```
const basic_string< _CharT, _Traits, _Alloc > & __lhs,
_CharT __rhs ) [inline], [constexpr]
```

Concatenate string and character.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

operator+() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline], [constexpr]
```

Concatenate string and C string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

operator+() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [constexpr]
```

Concatenate two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

operator<() [1/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator< (
```

```
const _CharT * __lhs,
const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string precedes string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

operator<() [2/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator< (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string precedes C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

operator<() [3/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator< (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [noexcept]
```

Test if string precedes string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

operator<() [4/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator< (
```

```
const deque< _Tp, _Alloc > & __x,
const deque< _Tp, _Alloc > & __y ) [inline]
```

Deque ordering relation.

Parameters

\leftrightarrow __x	A deque.
\leftrightarrow __y	A deque of the same type as __x.

Returns

True iff *x* is lexicographically less than __y.

This is a total ordering relation. It is linear in the size of the deques. The elements must be comparable with <. See `std::lexicographical_compare()` for how the determination is made.

References `std::deque< _Tp, _Alloc >::begin()`, and `std::deque< _Tp, _Alloc >::end()`.

operator<() [5/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator< (
    const forward_list< _Tp, _Alloc > & __lx,
    const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Forward list ordering relation.

Parameters

\leftrightarrow __lx	A forward_list.
\leftrightarrow __ly	A forward_list of the same type as __lx.

Returns

True iff __lx is lexicographically less than __ly.

This is a total ordering relation. It is linear in the number of elements of the forward lists. The elements must be comparable with <.

See `std::lexicographical_compare()` for how the determination is made.

References `std::forward_list< _Tp, _Alloc >::cbegin()`, and `std::forward_list< _Tp, _Alloc >::cend()`.

operator<() [6/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator< (
    const list< _Tp, _Alloc > & __x,
    const list< _Tp, _Alloc > & __y ) [inline]
```

List ordering relation.

Parameters

\leftrightarrow __x	A list.
--------------------------	---------

Parameters

$_x$	A list of the same type as $_x$.
$_y$	

Returns

True iff $_x$ is lexicographically less than $_y$.

This is a total ordering relation. It is linear in the size of the lists. The elements must be comparable with $<$. See `std::lexicographical_compare()` for how the determination is made.

References [std::list<_Tp, _Alloc>::begin\(\)](#), and [std::list<_Tp, _Alloc>::end\(\)](#).

operator<() [7/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator< (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Map ordering relation.

Parameters

$_x$	A map.
$_y$	A map of the same type as x .

Returns

True iff x is lexicographically less than y .

This is a total ordering relation. It is linear in the size of the maps. The elements must be comparable with $<$. See `std::lexicographical_compare()` for how the determination is made.

operator<() [8/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator< (
    const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Multimap ordering relation.

Parameters

$_x$	A multimap.
$_y$	A multimap of the same type as $_x$.

Returns

True iff x is lexicographically less than y .

This is a total ordering relation. It is linear in the size of the multimaps. The elements must be comparable with $<$. See `std::lexicographical_compare()` for how the determination is made.

operator<() [9/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator< (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Multiset ordering relation.

Parameters

\leftrightarrow _x	A multiset.
\leftrightarrow _y	A multiset of the same type as __x.

Returns

True iff __x is lexicographically less than __y.

This is a total ordering relation. It is linear in the size of the sets. The elements must be comparable with $<$. See `std::lexicographical_compare()` for how the determination is made.

operator<() [10/13]

```
template<typename _Tp , typename _Seq >
bool std::operator< (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Queue ordering relation.

Parameters

\leftrightarrow _x	A queue.
\leftrightarrow _y	A queue of the same type as x.

Returns

True iff __x is lexicographically less than __y.

This is an total ordering relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, the elements must be comparable with $<$, and `std::lexicographical_compare()` is usually used to make the determination.

References [std::queue< _Tp, _Sequence >::c](#).

operator<() [11/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
```

```
bool std::operator< (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Set ordering relation.

Parameters

\leftrightarrow __x	A set.
\leftrightarrow __y	A set of the same type as x.

Returns

True iff __x is lexicographically less than __y.

This is a total ordering relation. It is linear in the size of the sets. The elements must be comparable with <. See std::lexicographical_compare() for how the determination is made.

operator<() [12/13]

```
template<typename _Tp , typename _Seq >
bool std::operator< (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Stack ordering relation.

Parameters

\leftrightarrow __x	A stack.
\leftrightarrow __y	A stack of the same type as x.

Returns

True iff x is lexicographically less than __y.

This is an total ordering relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, the elements must be comparable with <, and std::lexicographical_compare() is usually used to make the determination.

operator<() [13/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator< (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline]
```

Vector ordering relation.

Parameters

\leftrightarrow __x	A vector.
--------------------------	-----------

Parameters

<code>__x</code>	A vector of the same type as <code>__x</code> .
<code>__y</code>	

Returns

True iff `__x` is lexicographically less than `__y`.

This is a total ordering relation. It is linear in the size of the vectors. The elements must be comparable with `<`. See `std::lexicographical_compare()` for how the determination is made.

References `std::vector<_Tp, _Alloc>::begin()`, and `std::vector<_Tp, _Alloc>::end()`.

operator<<() [1/24]

```
template<typename _Ostream , typename _Tp >
__rvalue_stream_insertion_t< _Ostream, _Tp > std::operator<< (
    _Ostream && __os,
    const _Tp & __x ) [inline]
```

Generic inserter for rvalue stream.

Parameters

<code>__os</code>	An input stream.
<code>__x</code>	A reference to the object being inserted.

Returns

`__os`

This is just a forwarding function to allow insertion to rvalue streams since they won't bind to the inserter functions that take an lvalue reference.

operator<<() [2/24]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __os,
    const __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Write string to a stream.

Parameters

<code>__os</code>	Output stream.
<code>__str</code>	String to write out.

Returns

Reference to the output stream.

Output characters of `__str` into `os` following the same rules as for writing a C string.

operator<<() [3/24]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __os,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Write string to a stream.

Parameters

<code>__os</code>	Output stream.
<code>__str</code>	String to write out.

Returns

Reference to the output stream.

Output characters of `__str` into `os` following the same rules as for writing a C string.

operator<<() [4/24]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __out,
    _CharT __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [5/24]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __out,
    char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [6/24]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __out,
    const _CharT * __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [7/24]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (
    basic_ostream< _CharT, _Traits > & __out,
    const char * __s )
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

___s must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(___s)` characters starting at `___s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

operator<<() [8/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & ,
    char16_t ) [delete]
```

Character inserters.

Parameters

___out	An output stream.
___c	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

If `___c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [9/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & ,
    char32_t ) [delete]
```

Character inserters.

Parameters

___out	An output stream.
___c	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

If `___c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [10/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & ,
    const char16_t * ) [delete]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [11/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & ,
    const char32_t * ) [delete]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [12/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
```



```
basic_ostream< char, _Traits > & ,
const wchar_t * ) [delete]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [13/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & ,
    wchar_t ) [delete]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [14/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [15/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    const char * __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [16/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    const signed char * __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [17/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    const unsigned char * __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [18/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    signed char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [19/24]

```
template<typename _Traits >
basic_ostream< char, _Traits > & std::operator<< (
    basic_ostream< char, _Traits > & __out,
    unsigned char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [20/24]

```
template<typename _Traits >
basic_ostream< wchar_t, _Traits > & std::operator<< (
    basic_ostream< wchar_t, _Traits > & ,
    char16_t ) [delete]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [21/24]

```
template<typename _Traits >
basic_ostream< wchar_t, _Traits > & std::operator<< (
    basic_ostream< wchar_t, _Traits > & ,
    char32_t ) [delete]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

operator<<() [22/24]

```
template<typename _Traits >
basic_ostream< wchar_t, _Traits > & std::operator<< (
    basic_ostream< wchar_t, _Traits > & ,
    const char16_t * ) [delete]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [23/24]

```
template<typename _Traits >
basic_ostream< wchar_t, _Traits > & std::operator<< (
    basic_ostream< wchar_t, _Traits > & ,
    const char32_t * ) [delete]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

operator<<() [24/24]

```
template<class _CharT , class _Traits , size_t _Nb>
std::basic_ostream< _CharT, _Traits > & std::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const bitset< _Nb > & __x )
```

Global I/O operators for bitsets.

Direct I/O between streams and bitsets is supported. Output is straightforward. Input will skip whitespace, only accept 0 and 1 characters, and will only extract as many digits as the bitset will hold.

Referenced by `std::shared_ptr< _Tp >::operator<<()`.

operator<=() [1/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator<= (
    const _CharT * __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string doesn't follow string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

operator<=() [2/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator<= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string doesn't follow C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

operator<=() [3/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator<= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [noexcept]
```

Test if string doesn't follow string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

operator<=() [4/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator<= (
    const deque< _Tp, _Alloc > & __x,
    const deque< _Tp, _Alloc > & __y ) [inline]
```

Based on `operator<`.

operator<=() [5/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator<= (
    const forward_list< _Tp, _Alloc > & __lx,
    const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Based on `operator<`.

operator<=() [6/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator<= (
    const list< _Tp, _Alloc > & __x,
    const list< _Tp, _Alloc > & __y ) [inline]
```

Based on `operator<`.

operator<=() [7/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator<= (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on `operator<`.

operator<=() [8/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator<= (
```

```
const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator<.

operator<=() [9/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator<= (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(y < x)

operator<=() [10/13]

```
template<typename _Tp , typename _Seq >
bool std::operator<= (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator<=() [11/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator<= (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(y < x)

operator<=() [12/13]

```
template<typename _Tp , typename _Seq >
bool std::operator<= (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator<=() [13/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator<= (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator==() [1/15]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator==(
    const _CharT * __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test equivalence of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) == 0`. False otherwise.

operator==() [2/15]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr bool std::operator== (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline], [constexpr]
```

Test equivalence of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

operator==() [3/15]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr bool std::operator== (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [constexpr],
[noexcept]
```

Test equivalence of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

operator==() [4/15]

```
template<typename _Tp , typename _Alloc >
bool std::operator== (
    const deque< _Tp, _Alloc > & __x,
    const deque< _Tp, _Alloc > & __y ) [inline]
```

Deque equality comparison.

Parameters

<code>__x</code>	A deque.
<code>__y</code>	A deque of the same type as <code>__x</code> .

Returns

True iff the size and elements of the deque are equal.

This is an equivalence relation. It is linear in the size of the deque. Deques are considered equivalent if their sizes are equal, and if corresponding elements compare equal.



References [std::deque<_Tp, _Alloc>::begin\(\)](#), [std::deque<_Tp, _Alloc>::end\(\)](#), and [std::deque<_Tp, _Alloc>::size\(\)](#).

operator==([5/15]

```
template<typename _Tp, typename _Alloc >
bool std::operator==(
    const forward_list<_Tp, _Alloc > & __lx,
    const forward_list<_Tp, _Alloc > & __ly )
```

Forward list equality comparison.

Parameters

 _lx	A forward_list
 _ly	A forward_list of the same type as _lx .

Returns

True iff the elements of the forward lists are equal.

This is an equivalence relation. It is linear in the number of elements of the forward lists. Deques are considered equivalent if corresponding elements compare equal.

References [std::forward_list<_Tp, _Alloc>::cbegin\(\)](#), and [std::forward_list<_Tp, _Alloc>::cend\(\)](#).

operator==([6/15]

```
template<typename _StateT >
bool std::operator==(
    const fpos<_StateT > & __lhs,
    const fpos<_StateT > & __rhs ) [inline]
```

Test if equivalent to another position.

operator==([7/15]

```
template<typename _Res, typename... _Args>
bool std::operator==(
    const function<_Res(_Args...)> & __f,
    nullptr_t ) [inline], [noexcept]
```

Test whether a polymorphic function object wrapper is empty.

Returns

true if the wrapper has no target, false otherwise

This function will not throw exceptions.

operator==([8/15]

```
template<typename _Tp, typename _Alloc >
bool std::operator==(
```

```
const list< _Tp, _Alloc > & __x,
const list< _Tp, _Alloc > & __y ) [inline]
```

List equality comparison.

Parameters

\leftrightarrow _x	A list.
\leftrightarrow _y	A list of the same type as __x.

Returns

True iff the size and elements of the lists are equal.

This is an equivalence relation. It is linear in the size of the lists. Lists are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

References `std::list< _Tp, _Alloc >::begin()`, `std::list< _Tp, _Alloc >::end()`, and `std::list< _Tp, _Alloc >::size()`.

operator==() [9/15]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator== (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Map equality comparison.

Parameters

\leftrightarrow _x	A map.
\leftrightarrow _y	A map of the same type as x.

Returns

True iff the size and elements of the maps are equal.

This is an equivalence relation. It is linear in the size of the maps. Maps are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

operator==() [10/15]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator== (
    const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Multimap equality comparison.

Parameters

\leftrightarrow _x	A multimap.
\leftrightarrow _y	A multimap of the same type as __x.

Returns

True iff the size and elements of the maps are equal.

This is an equivalence relation. It is linear in the size of the multimaps. Multimaps are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

operator==() [11/15]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator== (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Multiset equality comparison.

Parameters

\leftrightarrow __x	A multiset.
\leftrightarrow __y	A multiset of the same type as __x.

Returns

True iff the size and elements of the multisets are equal.

This is an equivalence relation. It is linear in the size of the multisets. Multisets are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

operator==() [12/15]

```
template<typename _Tp , typename _Seq >
bool std::operator== (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Queue equality comparison.

Parameters

\leftrightarrow __x	A queue.
\leftrightarrow __y	A queue of the same type as __x.

Returns

True iff the size and elements of the queues are equal.

This is an equivalence relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, and queues are considered equivalent if their sequences compare equal.

References [std::queue< _Tp, _Sequence >::c](#).

operator==() [13/15]

```
template<typename _Key , typename _Compare , typename _Alloc >
```

```
bool std::operator== (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Set equality comparison.

Parameters

\leftrightarrow __x	A set.
\leftrightarrow __y	A set of the same type as x.

Returns

True iff the size and elements of the sets are equal.

This is an equivalence relation. It is linear in the size of the sets. Sets are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

operator==() [14/15]

```
template<typename _Tp , typename _Seq >
bool std::operator== (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Stack equality comparison.

Parameters

\leftrightarrow __x	A stack.
\leftrightarrow __y	A stack of the same type as __x.

Returns

True iff the size and elements of the stacks are equal.

This is an equivalence relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, and stacks are considered equivalent if their sequences compare equal.

operator==() [15/15]

```
template<typename _Tp , typename _Alloc >
constexpr bool std::operator== (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline], [constexpr]
```

Vector equality comparison.

Parameters

\leftrightarrow __x	A vector.
--------------------------	-----------

Parameters

<code>__x</code>	A vector of the same type as <code>__x</code> .
<code>__y</code>	

Returns

True iff the size and elements of the vectors are equal.

This is an equivalence relation. It is linear in the size of the vectors. Vectors are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

References [std::vector<_Tp, _Alloc>::begin\(\)](#), [std::vector<_Tp, _Alloc>::end\(\)](#), and [std::vector<_Tp, _Alloc>::size\(\)](#).

operator>() [1/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator> (
    const _CharT * __lhs,
    const basic\_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string follows string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

operator>() [2/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator> (
    const basic\_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string follows C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

operator>() [3/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator> (
```

```
const basic_string< _CharT, _Traits, _Alloc > & __lhs,
const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [noexcept]
```

Test if string follows string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

operator>() [4/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator> (
    const deque< _Tp, _Alloc > & __x,
    const deque< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>() [5/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator> (
    const forward_list< _Tp, _Alloc > & __lx,
    const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Based on operator<.

operator>() [6/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator> (
    const list< _Tp, _Alloc > & __x,
    const list< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>() [7/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator> (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>() [8/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator> (
    const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>() [9/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator> (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns $y < x$.

operator>() [10/13]

```
template<typename _Tp , typename _Seq >
bool std::operator> (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator>() [11/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator> (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns $y < x$.

operator>() [12/13]

```
template<typename _Tp , typename _Seq >
bool std::operator> (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator>() [13/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator> (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>=() [1/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator>= (
    const _CharT * __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string doesn't precede string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

operator>=() [2/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator>= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const _CharT * __rhs ) [inline]
```

Test if string doesn't precede C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

operator>=() [3/13]

```
template<typename _CharT , typename _Traits , typename _Alloc >
bool std::operator>= (
    const basic_string< _CharT, _Traits, _Alloc > & __lhs,
    const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [noexcept]
```

Test if string doesn't precede string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

operator>=() [4/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator>= (
    const deque< _Tp, _Alloc > & __x,
    const deque< _Tp, _Alloc > & __y ) [inline]
```

Based on `operator<`.

operator>=() [5/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator>= (
    const forward_list< _Tp, _Alloc > & __lx,
    const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Based on `operator<`.

operator>=() [6/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator>= (
    const list< _Tp, _Alloc > & __x,
    const list< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>=() [7/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator>= (
    const map< _Key, _Tp, _Compare, _Alloc > & __x,
    const map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>=() [8/13]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
bool std::operator>= (
    const multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    const multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>=() [9/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator>= (
    const multiset< _Key, _Compare, _Alloc > & __x,
    const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(x < y)

operator>=() [10/13]

```
template<typename _Tp , typename _Seq >
bool std::operator>= (
    const queue< _Tp, _Seq > & __x,
    const queue< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator>=() [11/13]

```
template<typename _Key , typename _Compare , typename _Alloc >
bool std::operator>= (
    const set< _Key, _Compare, _Alloc > & __x,
    const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns !(x < y)

operator>=() [12/13]

```
template<typename _Tp , typename _Seq >
bool std::operator>= (
    const stack< _Tp, _Seq > & __x,
    const stack< _Tp, _Seq > & __y ) [inline]
```

Based on operator<.

operator>=() [13/13]

```
template<typename _Tp , typename _Alloc >
bool std::operator>= (
    const vector< _Tp, _Alloc > & __x,
    const vector< _Tp, _Alloc > & __y ) [inline]
```

Based on operator<.

operator>>() [1/10]

```
template<typename _Istream , typename _Tp >
__rvalue_stream_extraction_t< _Istream, _Tp > std::operator>> (
    _Istream && __is,
    _Tp && __x ) [inline]
```

Generic extractor for rvalue stream.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A reference to the extraction target.

Returns

`__is`

This is just a forwarding function to allow extraction from rvalue streams since they won't bind to the extractor functions that take an lvalue reference.

operator>>() [2/10]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::operator>> (
    basic_istream< _CharT, _Traits > & __in,
    _CharT & __c )
```

Character extractors.

Parameters

<code>__in</code>	An input stream.
<code>__c</code>	A character reference.

Returns

`in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

References `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rd` and `std::basic_ios< _CharT, _Traits >::setstate()`.

operator>>() [3/10]

```
template<typename _CharT , typename _Traits , size_t _Num>
basic_istream< _CharT, _Traits > & std::operator>> (
    basic_istream< _CharT, _Traits > & __in,
    _CharT(&) __s[_Num] ) [inline]
```

Character string extractors.

Parameters

<code>__in</code>	An input stream.
<code>__s</code>	A character array (or a pointer to an array before C++20).

Returns

`__in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `min(width(), n)`
- otherwise `n` is the number of elements of the array
- (before C++20 the pointer is assumed to point to an array of the largest possible size for an array of `char_type`).

Characters are extracted and stored until one of the following happens:

- `n - 1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

operator>>() [4/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
basic_istream< _CharT, _Traits > & std::operator>> (
    basic_istream< _CharT, _Traits > & __is,
    __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str )
```

Read stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until whitespace is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased.

References `std::ios_base::getloc()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::ios_base::width()`.

operator>>() [5/10]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::operator>> (
    basic_istream<_CharT, _Traits> & __is,
    basic_string<_CharT, _Traits, _Alloc> & __str )
```

Read stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until whitespace is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::erase()`, `std::ios_base::getloc()`, `std::basic_string<_CharT, _Traits, _Alloc>::max_size()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::ios_base::width()`.

operator>>() [6/10]

```
template<class _Traits>
basic_istream<char, _Traits> & std::operator>> (
    basic_istream<char, _Traits> & __in,
    signed char & __c ) [inline]
```

Character extractors.

Parameters

<code>__in</code>	An input stream.
<code>__c</code>	A character reference.

Returns

`in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

operator>>() [7/10]

```
template<class _Traits , size_t _Num>
basic_istream< char, _Traits > & std::operator>> (
    basic_istream< char, _Traits > & __in,
    signed char(&) __s[_Num] ) [inline]
```

Character string extractors.

Parameters

\leftarrow __in	An input stream.
\leftarrow __s	A character array (or a pointer to an array before C++20).

Returns

__in

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `min(width(), n)`
- otherwise `n` is the number of elements of the array
- (before C++20 the pointer is assumed to point to an array of the largest possible size for an array of `char_type`).

Characters are extracted and stored until one of the following happens:

- `n - 1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

operator>>() [8/10]

```
template<class _Traits >
basic_istream< char, _Traits > & std::operator>> (
    basic_istream< char, _Traits > & __in,
    unsigned char & __c ) [inline]
```

Character extractors.

Parameters

\leftarrow __in	An input stream.
\leftarrow __c	A character reference.

Returns`in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

operator>>() [9/10]

```
template<class _Traits , size_t _Num>
basic_istream< char, _Traits > & std::operator>> (
    basic_istream< char, _Traits > & __in,
    unsigned char(&) __s[_Num] ) [inline]
```

Character string extractors.

Parameters

<code>__in</code>	An input stream.
<code>__s</code>	A character array (or a pointer to an array before C++20).

Returns`__in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `min(width(), n)`
- otherwise `n` is the number of elements of the array
- (before C++20 the pointer is assumed to point to an array of the largest possible size for an array of `char_type`).

Characters are extracted and stored until one of the following happens:

- `n - 1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

operator>>() [10/10]

```
template<class _CharT , class _Traits , size_t _Nb>
std::basic_istream< _CharT, _Traits > & std::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    bitset< _Nb > & __x )
```

Global I/O operators for bitsets.

Direct I/O between streams and bitsets is supported. Output is straightforward. Input will skip whitespace, only accept 0 and 1 characters, and will only extract as many digits as the bitset will hold.

operator^()

```
template<size_t _Nb>
bitset< _Nb > std::operator^ (
    const bitset< _Nb > & __x,
    const bitset< _Nb > & __y ) [inline], [noexcept]
```

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

A new bitset.

These should be self-explanatory.

operator" | ()

```
template<size_t _Nb>
bitset< _Nb > std::operator| (
    const bitset< _Nb > & __x,
    const bitset< _Nb > & __y ) [inline], [noexcept]
```

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

A new bitset.

These should be self-explanatory.

put_money()

```
template<typename _MoneyT >
_Put_money< _MoneyT > std::put_money (
    const _MoneyT & __mon,
    bool __intl = false ) [inline]
```

Extended manipulator for inserting money.

Parameters

$_mon$	Either long double or a specialization of <code>basic_string</code> .
$_intl$	A bool indicating whether international format is to be used.

Sent to a stream object, this manipulator inserts `__mon`.

put_time()

```
template<typename _CharT >
_Put_time< _CharT > std::put_time (
    const std::tm * __tmb,
    const _CharT * __fmt ) [inline]
```

Extended manipulator for formatting time.

This manipulator uses `time_put::put` to format time. [ext.manip]

Parameters

<code>__tmb</code>	struct tm time data to format.
<code>__fmt</code>	format string.

quoted()

```
template<typename _CharT >
auto std::quoted (
    const _CharT * __string,
    _CharT __delim = _CharT('\''),
    _CharT __escape = _CharT('\\') ) [inline]
```

Manipulator for quoted strings.

Parameters

<code>__string</code>	String to quote.
<code>__delim</code>	Character to quote string with.
<code>__escape</code>	Escape character to escape itself or quote character.

rbegin() [1/4]

```
template<typename _Container >
constexpr auto std::rbegin (
    _Container & __cont ) -> decltype(__cont.rbegin()) [inline], [constexpr]
```

Return a reverse iterator pointing to the last element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Referenced by [crbegin\(\)](#).

rbegin() [2/4]

```
template<typename _Tp , size_t _Nm>
constexpr reverse\_iterator< _Tp * > std::rbegin (
    _Tp(&) __arr[_Nm] ) [inline], [constexpr], [noexcept]
```

Return a reverse iterator pointing to the last element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

rbegin() [3/4]

```
template<typename _Container >
constexpr auto std::rbegin (
    const _Container & __cont ) -> decltype(__cont.rbegin())    [inline], [constexpr]
```

Return a reverse iterator pointing to the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

rbegin() [4/4]

```
template<typename _Tp >
constexpr reverse_iterator< const _Tp * > std::rbegin (
    initializer_list< _Tp > __il ) [inline], [constexpr], [noexcept]
```

Return a reverse iterator pointing to the last element of the initializer_list.

Parameters

<code>__il</code>	initializer_list.
-------------------	-------------------

rend() [1/4]

```
template<typename _Container >
constexpr auto std::rend (
    _Container & __cont ) -> decltype(__cont.rend())    [inline], [constexpr]
```

Return a reverse iterator pointing one past the first element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Referenced by [crend\(\)](#).

rend() [2/4]

```
template<typename _Tp , size_t _Nm>
constexpr reverse_iterator< _Tp * > std::rend (
    _Tp(&) __arr[_Nm] ) [inline], [constexpr], [noexcept]
```

Return a reverse iterator pointing one past the first element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

rend() [3/4]

```
template<typename _Container >
constexpr auto std::rend (
    const _Container & __cont ) -> decltype(__cont.rend())    [inline], [constexpr]
```

Return a reverse iterator pointing one past the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

rend() [4/4]

```
template<typename _Tp >
constexpr reverse_iterator< const _Tp * > std::rend (
    initializer_list< _Tp > __il ) [inline], [constexpr], [noexcept]
```

Return a reverse iterator pointing one past the first element of the initializer_list.

Parameters

<code>__il</code>	initializer_list.
-------------------	-------------------

replace_copy()

```
template<typename _InputIterator , typename _OutputIterator , typename _Tp >
constexpr _OutputIterator std::replace_copy (
    _InputIterator __first,
    _InputIterator __last,
    _OutputIterator __result,
    const _Tp & __old_value,
    const _Tp & __new_value ) [inline], [constexpr]
```

Copy a sequence, replacing each element of one value with another value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__old_value</code>	The value to be replaced.
<code>__new_value</code>	The replacement value.

Returns

The end of the output sequence, `result+(last-first)`.

Copies each element in the input range [`__first`, `__last`) to the output range [`__result`, `__result+(__last-__first)`) replacing elements equal to `__old_value` with `__new_value`.

resetiosflags()

```
_Resetiosflags std::resetiosflags (
    ios_base::fmtflags __mask ) [inline]
```

Manipulator for `setf`.

Parameters

<code>__mask</code>	A format flags mask.
---------------------	----------------------

Sent to a stream object, this manipulator resets the specified flags, via `stream.setf(0,__mask)`.

return_temporary_buffer()

```
template<typename _Tp >
void std::return_temporary_buffer (
    _Tp * __p ) [inline]
```

The companion to `get_temporary_buffer()`.

Parameters

<code>__p</code>	A buffer previously allocated by <code>get_temporary_buffer</code> .
------------------	--

Returns

None.

Frees the memory pointed to by `__p`.

right()

```
ios_base & std::right (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::right, ios_base::adjustfield)`.

References `std::ios_base::adjustfield`, and `std::ios_base::right`.

sample()

```
template<typename _PopulationIterator , typename _SampleIterator , typename _Distance , typename
_UniformRandomBitGenerator >
_SampleIterator std::sample (
    _PopulationIterator __first,
    _PopulationIterator __last,
    _SampleIterator __out,
    _Distance __n,
    _UniformRandomBitGenerator && __g )
```

Take a random sample from a population.

scientific()

```
ios_base & std::scientific (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::scientific, ios_base::floatfield)`.

References `std::ios_base::floatfield`, `std::ios_base::scientific`, and `scientific()`.

Referenced by `scientific()`.

search()

```
template<typename _ForwardIterator , typename _Searcher >
constexpr _ForwardIterator std::search (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Searcher & __searcher ) [inline], [constexpr]
```

Search a sequence using a Searcher object.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__searcher</code>	A callable object.

Returns

```
__searcher(__first,__last).first
```

set_new_handler()

```
new_handler std::set_new_handler (
    new_handler ) throw ( )
```

Takes a replacement handler as the argument, returns the previous handler.

setbase()

```
_Setbase std::setbase (
    int __base ) [inline]
```

Manipulator for `setf`.

Parameters

<code>__base</code>	A numeric base.
---------------------	-----------------

Sent to a stream object, this manipulator changes the `ios_base::basefield` flags to `oct`, `dec`, or `hex` when `base` is 8, 10, or 16, accordingly, and to 0 if `__base` is any other value.

setfill()

```
template<typename _CharT >
_Setfill< _CharT > std::setfill (
    _CharT __c ) [inline]
```

Manipulator for `fill`.

Parameters

<code>__c</code>	The new fill character.
------------------	-------------------------

Sent to a stream object, this manipulator calls `fill(__c)` for that object.

setiosflags()

```
_Setiosflags std::setiosflags (
    ios_base::fmtflags __mask ) [inline]
```

Manipulator for `setf`.

Parameters

<code>__mask</code>	A format flags mask.
---------------------	----------------------

Sent to a stream object, this manipulator sets the format flags to `__mask`.

setprecision()

```
_Setprecision std::setprecision (
    int __n ) [inline]
```

Manipulator for `precision`.

Parameters

<code>__n</code>	The new precision.
------------------	--------------------

Sent to a stream object, this manipulator calls `precision(__n)` for that object.

setw()

```
_Setw std::setw (
    int __n ) [inline]
```

Manipulator for `width`.

Parameters

<code>__n</code>	The new width.
------------------	----------------

Sent to a stream object, this manipulator calls `width(__n)` for that object.

showbase()

```
ios_base & std::showbase (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::showbase)`.

References [std::ios_base::showbase](#).

showpoint()

```
ios_base & std::showpoint (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::showpoint)`.

References [std::ios_base::showpoint](#).

showpos()

```
ios_base & std::showpos (
```

```
ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::showpos)`.

References [std::ios_base::showpos](#).

size() [1/2]

```
template<typename _Container >
constexpr auto std::size (
    const _Container & __cont ) -> decltype(__cont.size()) [constexpr], [noexcept]
```

Return the size of a container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Referenced by [std::deque<_Tp, _Alloc>::_M_new_elements_at_back\(\)](#), [std::deque<_Tp, _Alloc>::_M_new_elements_at_front\(\)](#), [std::deque<_Tp, _Alloc>::_M_push_back_aux\(\)](#), [std::deque<_Tp, _Alloc>::_M_push_front_aux\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), [std::deque<_Tp, _Alloc>::operator=\(\)](#), [std::vector<_Tp, _Alloc>::operator=\(\)](#), [std::vector<_Tp, _Alloc>::reserve\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::reserve\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#).

size() [2/2]

```
template<typename _Tp , size_t _Nm>
constexpr size_t std::size (
    const _Tp(&)[_Nm] ) [constexpr], [noexcept]
```

Return the size of an array.

skipws()

```
ios_base & std::skipws (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::skipws)`.

References [std::ios_base::skipws](#).

static_pointer_cast()

```
template<typename _Tp , typename _Tp1 , _Lock_policy _Lp>
__shared_ptr<_Tp, _Lp> std::static_pointer_cast (
    const __shared_ptr<_Tp1, _Lp> & __r ) [inline], [noexcept]
static_pointer_cast
```

swap() [1/18]

```
template<class _CharT , class _Traits >
void std::swap (
    basic_filebuf<_CharT, _Traits> & __x,
    basic_filebuf<_CharT, _Traits> & __y ) [inline]
```

Swap specialization for filebufs.

swap() [2/18]

```
template<class _CharT , class _Traits >
void std::swap (
    basic_fstream< _CharT, _Traits > & __x,
    basic_fstream< _CharT, _Traits > & __y ) [inline]
```

Swap specialization for fstreams.

swap() [3/18]

```
template<class _CharT , class _Traits >
void std::swap (
    basic_ifstream< _CharT, _Traits > & __x,
    basic_ifstream< _CharT, _Traits > & __y ) [inline]
```

Swap specialization for ifstreams.

swap() [4/18]

```
template<class _CharT , class _Traits , class _Allocator >
void std::swap (
    basic_istringstream< _CharT, _Traits, _Allocator > & __x,
    basic_istringstream< _CharT, _Traits, _Allocator > & __y ) [inline]
```

Swap specialization for istringstreams.

swap() [5/18]

```
template<class _CharT , class _Traits >
void std::swap (
    basic_ofstream< _CharT, _Traits > & __x,
    basic_ofstream< _CharT, _Traits > & __y ) [inline]
```

Swap specialization for ofstreams.

swap() [6/18]

```
template<class _CharT , class _Traits , class _Allocator >
void std::swap (
    basic_ostringstream< _CharT, _Traits, _Allocator > & __x,
    basic_ostringstream< _CharT, _Traits, _Allocator > & __y ) [inline]
```

Swap specialization for ostringstreams.

swap() [7/18]

```
template<typename _CharT , typename _Traits , typename _Alloc >
constexpr void std::swap (
    basic_string< _CharT, _Traits, _Alloc > & __lhs,
    basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline], [constexpr], [noexcept]
```

Swap contents of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Exchanges the contents of `__lhs` and `__rhs` in constant time.

swap() [8/18]

```
template<class _CharT , class _Traits , class _Allocator >
void std::swap (
    basic_stringbuf< _CharT, _Traits, _Allocator > & __x,
    basic_stringbuf< _CharT, _Traits, _Allocator > & __y ) [inline], [noexcept]
```

Swap specialization for stringbufs.

swap() [9/18]

```
template<class _CharT , class _Traits , class _Allocator >
void std::swap (
    basic_stringstream< _CharT, _Traits, _Allocator > & __x,
    basic_stringstream< _CharT, _Traits, _Allocator > & __y ) [inline]
```

Swap specialization for stringstream.

swap() [10/18]

```
template<typename _Tp , typename _Alloc >
void std::swap (
    deque< _Tp, _Alloc > & __x,
    deque< _Tp, _Alloc > & __y ) [inline], [noexcept]
```

See std::deque::swap().

swap() [11/18]

```
template<typename _Tp , typename _Alloc >
void std::swap (
    forward_list< _Tp, _Alloc > & __lx,
    forward_list< _Tp, _Alloc > & __ly ) [inline], [noexcept]
```

See std::forward_list::swap().

swap() [12/18]

```
template<typename _Res , typename... _Args>
void std::swap (
    function< _Res(_Args...)> & __x,
    function< _Res(_Args...)> & __y ) [inline], [noexcept]
```

Swap the targets of two polymorphic function object wrappers.

This function will not throw exceptions.

swap() [13/18]

```
template<typename _Tp , typename _Alloc >
void std::swap (
    list< _Tp, _Alloc > & __x,
    list< _Tp, _Alloc > & __y ) [inline], [noexcept]
```

See std::list::swap().

swap() [14/18]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
void std::swap (
    map< _Key, _Tp, _Compare, _Alloc > & __x,
    map< _Key, _Tp, _Compare, _Alloc > & __y ) [inline], [noexcept]
```

See `std::map::swap()`.

References [std::pair<_T1, _T2>::swap\(\)](#).

swap() [15/18]

```
template<typename _Key , typename _Tp , typename _Compare , typename _Alloc >
void std::swap (
    multimap< _Key, _Tp, _Compare, _Alloc > & __x,
    multimap< _Key, _Tp, _Compare, _Alloc > & __y ) [inline], [noexcept]
```

See `std::multimap::swap()`.

References [std::pair<_T1, _T2>::swap\(\)](#).

swap() [16/18]

```
template<typename _Key , typename _Compare , typename _Alloc >
void std::swap (
    multiset< _Key, _Compare, _Alloc > & __x,
    multiset< _Key, _Compare, _Alloc > & __y ) [inline], [noexcept]
```

See `std::multiset::swap()`.

swap() [17/18]

```
template<typename _Key , typename _Compare , typename _Alloc >
void std::swap (
    set< _Key, _Compare, _Alloc > & __x,
    set< _Key, _Compare, _Alloc > & __y ) [inline], [noexcept]
```

See `std::set::swap()`.

swap() [18/18]

```
template<typename _Tp , typename _Alloc >
constexpr void std::swap (
    vector< _Tp, _Alloc > & __x,
    vector< _Tp, _Alloc > & __y ) [inline], [constexpr], [noexcept]
```

See `std::vector::swap()`.

tolower()

```
template<typename _CharT >
_CharT std::tolower (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype::tolower(__c)`.

References [tolower\(\)](#).

Referenced by [tolower\(\)](#).

toupper()

```
template<typename _CharT >
_CharT std::toupper (
    _CharT __c,
    const locale & __loc ) [inline]
```

Convenience interface to `ctype::toupper(__c)`.

References [toupper\(\)](#).

Referenced by [toupper\(\)](#).

unitbuf()

```
ios_base & std::unitbuf (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::unitbuf)`.

References [std::ios_base::unitbuf](#).

uppercase()

```
ios_base & std::uppercase (
    ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::uppercase)`.

References [std::ios_base::uppercase](#).

ws()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::ws (
    basic_istream< _CharT, _Traits > & __is )
```

Quick and easy way to eat whitespace.

This manipulator extracts whitespace characters, stopping when the next character is non-whitespace, or when the input sequence is empty. If the sequence is empty, `eofbit` is set in the stream, but not `failbit`.

The current locale is used to distinguish whitespace characters.

Example:

```
MyClass mc;
```

```
std::cin >> std::ws >> mc;
```

will skip leading whitespace before calling `operator>>` on `cin` and your object. Note that the same effect can be achieved by creating a `std::basic_istream::sentry` inside your definition of `operator>>`.

References [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::getloc\(\)](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

4.10.5 Variable Documentation**__ioint**

```
ios_base::Init std::__ioint [static]
```

Linked to standard error (buffered)

cerr

```
ostream std::cerr [extern]
```

Linked to standard output.

cin

```
istream std::cin [extern]
```

Linked to standard input.

clog

```
ostream std::clog [extern]
```

Linked to standard error (unbuffered)

cout

`ostream` `std::cout` [extern]

Linked to standard input.

default_sentinel

`constexpr` `default_sentinel_t` `std::default_sentinel` [inline], [constexpr]

A default sentinel value.

destroying_delete

`constexpr` `destroying_delete_t` `std::destroying_delete` [inline], [constexpr]

Tag variable of type `destroying_delete_t`.

wcerr

`wostream` `std::wcerr` [extern]

Linked to standard output.

wcin

`wistream` `std::wcin` [extern]

Linked to standard error (buffered)

wclog

`wostream` `std::wclog` [extern]

Linked to standard error (unbuffered)

wcout

`wostream` `std::wcout` [extern]

Linked to standard input.

4.11 std::__debug Namespace Reference**Classes**

- class `bitset`
- class `deque`
- class `forward_list`
- class `list`
- class `map`
- class `multimap`
- class `multiset`
- class `set`
- class `unordered_map`
- class `unordered_multimap`
- class `unordered_multiset`
- class `unordered_set`
- class `vector`

Functions

- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
deque (`_InputIterator, _InputIterator, _Allocator=_Allocator()`) -> `deque<_ValT, _Allocator>`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>`
deque (`size_t, _Tp, _Allocator=_Allocator()`) -> `deque<_Tp, _Allocator>`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
forward_list (`_InputIterator, _InputIterator, _Allocator=_Allocator()`) -> `forward_list<_ValT, _Allocator>`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>`
forward_list (`size_t, _Tp, _Allocator=_Allocator()`) -> `forward_list<_Tp, _Allocator>`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
list (`_InputIterator, _InputIterator, _Allocator=_Allocator()`) -> `list<_ValT, _Allocator>`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>`
list (`size_t, _Tp, _Allocator=_Allocator()`) -> `list<_Tp, _Allocator>`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
map (`_InputIterator, _InputIterator, _Allocator`) -> `map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, less<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_val_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
map (`_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()`) -> `map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Compare, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
map (`initializer_list<pair<_Key, _Tp>>, _Allocator`) -> `map<_Key, _Tp, less<_Key>, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
map (`initializer_list<pair<_Key, _Tp>>, _Compare=_Compare(), _Allocator=_Allocator()`) -> `map<_Key, _Tp, _Compare, _Allocator>`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
multimap (`_InputIterator, _InputIterator, _Allocator`) -> `multimap<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, less<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_val_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
multimap (`_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()`) -> `multimap<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Compare, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
multimap (`initializer_list<pair<_Key, _Tp>>, _Allocator`) -> `multimap<_Key, _Tp, less<_Key>, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
multimap (`initializer_list<pair<_Key, _Tp>>, _Compare=_Compare(), _Allocator=_Allocator()`) -> `multimap<_Key, _Tp, _Compare, _Allocator>`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
multiset (`_InputIterator, _InputIterator, _Allocator`) -> `multiset<typename iterator_traits<_InputIterator>::value_type, less<typename iterator_traits<_InputIterator>::value_type>, _Allocator>`

- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
multiset (`_InputIterator`, `_InputIterator`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `multiset<typename iterator_traits<_InputIterator>::value_type, _Compare, _Allocator>`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
multiset (`initializer_list<_Key>`, `_Allocator`) -> `multiset<_Key, less<_Key>, _Allocator>`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
multiset (`initializer_list<_Key>`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `multiset<_Key, _Compare, _Allocator>`
- `template<typename _Tp, typename _Alloc>`
operator!= (`const deque<_Tp, _Alloc> &__lhs`, `const deque<_Tp, _Alloc> &__rhs`)
- `template<typename _Tp, typename _Alloc>`
operator!= (`const forward_list<_Tp, _Alloc> &__lx`, `const forward_list<_Tp, _Alloc> &__ly`)
- `template<typename _Tp, typename _Alloc>`
operator!= (`const list<_Tp, _Alloc> &__lhs`, `const list<_Tp, _Alloc> &__rhs`)
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator>`
operator!= (`const map<_Key, _Tp, _Compare, _Allocator> &__lhs`, `const map<_Key, _Tp, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator>`
operator!= (`const multimap<_Key, _Tp, _Compare, _Allocator> &__lhs`, `const multimap<_Key, _Tp, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Compare, typename _Allocator>`
operator!= (`const multiset<_Key, _Compare, _Allocator> &__lhs`, `const multiset<_Key, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Compare, typename _Allocator>`
operator!= (`const set<_Key, _Compare, _Allocator> &__lhs`, `const set<_Key, _Compare, _Allocator> &__rhs`)
- `template<typename _Tp, typename _Alloc>`
operator!= (`const vector<_Tp, _Alloc> &__lhs`, `const vector<_Tp, _Alloc> &__rhs`)
- `template<size_t _Nb>`
bitset<_Nb> operator& (`const bitset<_Nb> &__x`, `const bitset<_Nb> &__y`) noexcept
- `template<typename _Tp, typename _Alloc>`
operator< (`const deque<_Tp, _Alloc> &__lhs`, `const deque<_Tp, _Alloc> &__rhs`)
- `template<typename _Tp, typename _Alloc>`
operator< (`const forward_list<_Tp, _Alloc> &__lx`, `const forward_list<_Tp, _Alloc> &__ly`)
- `template<typename _Tp, typename _Alloc>`
operator< (`const list<_Tp, _Alloc> &__lhs`, `const list<_Tp, _Alloc> &__rhs`)
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator>`
operator< (`const map<_Key, _Tp, _Compare, _Allocator> &__lhs`, `const map<_Key, _Tp, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator>`
operator< (`const multimap<_Key, _Tp, _Compare, _Allocator> &__lhs`, `const multimap<_Key, _Tp, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Compare, typename _Allocator>`
operator< (`const multiset<_Key, _Compare, _Allocator> &__lhs`, `const multiset<_Key, _Compare, _Allocator> &__rhs`)
- `template<typename _Key, typename _Compare, typename _Allocator>`
operator< (`const set<_Key, _Compare, _Allocator> &__lhs`, `const set<_Key, _Compare, _Allocator> &__rhs`)
- `template<typename _Tp, typename _Alloc>`
operator< (`const vector<_Tp, _Alloc> &__lhs`, `const vector<_Tp, _Alloc> &__rhs`)

- `template<typename _CharT, typename _Traits, size_t _Nb>
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
bitset< _Nb > &__x)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,
_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator<= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >
&__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,
_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _
Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator== (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >
&__rhs)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map<
_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap<
_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset<
_Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value,
_Hash, _Pred, _Alloc > &__y)`

- `template<typename _Tp, typename _Alloc >`
`bool operator== (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, bitset< _Nb > &__x)`
- `template<size_t _Nb>`
`bitset< _Nb > operator^ (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>`
`bitset< _Nb > operator| (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`set (_InputIterator, _InputIterator, _Allocator) -> set< typename iterator_traits< _InputIterator >::value_type, less< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`

- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>>`
`set (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> set< typename iterator_traits< _InputIterator >::value_type, _Compare, _Allocator >`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`set (initializer_list< _Key >, _Allocator) -> set< _Key, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>>`
`set (initializer_list< _Key >, _Compare=_Compare(), _Allocator=_Allocator()) -> set< _Key, _Compare, _Allocator >`
- `template<typename _Tp, typename _Alloc >`
`void swap (deque< _Tp, _Alloc > &__lhs, deque< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _Tp, typename _Alloc >`
`void swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.swap(__ly)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (list< _Tp, _Alloc > &__lhs, list< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (set< _Key, _Compare, _Allocator > &__x, set< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (vector< _Tp, _Alloc > &__lhs, vector< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, _Allocator) -> unordered_map< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`unordered_map (_InputIterator, _InputIterator, typename unordered_map< int, int >::size_type, _Allocator) -> unordered_map< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >`

- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
unordered_map (`_InputIterator`, `_InputIterator`, `typename unordered_map< int, int >::size_type`, `_Hash`, `↵`
`_Allocator`) -> `unordered_map< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Hash,`
`equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash<__iter_key_t<_InputIterator>>, typename _Pred = equal_to<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_↵`
`Allocator>>>`
unordered_map (`_InputIterator`, `_InputIterator`, `typename unordered_map< int, int >::size_type={}`, `_Hash=↵`
`Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_map< __iter_key_t< _InputIterator >, __iter_↵`
`val_t< _InputIterator >, _Hash, _Pred, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
unordered_map (`initializer_list< pair< _Key, _Tp > >`, `_Allocator`) -> `unordered_map< _Key, _Tp, hash< _Key`
`>, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
unordered_map (`initializer_list< pair< _Key, _Tp > >`, `typename unordered_map< int, int >::size_type`, `↵`
`Allocator`) -> `unordered_map< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
unordered_map (`initializer_list< pair< _Key, _Tp > >`, `typename unordered_map< int, int >::size_type`, `_Hash,`
`_Allocator`) -> `unordered_map< _Key, _Tp, _Hash, equal_to< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
unordered_map (`initializer_list< pair< _Key, _Tp > >`, `typename unordered_map< int, int >::size_type={}`,
`_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_map< _Key, _Tp, _Hash, _Pred, ↵`
`Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`Allocator<_Allocator>>>`
unordered_multimap (`_InputIterator`, `_InputIterator`, `_Allocator`) -> `unordered_multimap< __iter_key_t< ↵`
`_InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_↵`
`key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`Allocator<_Allocator>>>`
unordered_multimap (`_InputIterator`, `_InputIterator`, `unordered_multimap< int, int >::size_type`, `_Allocator`) ->
`unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< ↵`
`_InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
unordered_multimap (`_InputIterator`, `_InputIterator`, `unordered_multimap< int, int >::size_type`, `_Hash, ↵`
`Allocator`) -> `unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Hash,`
`equal_to< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash<__iter_key_t<_InputIterator>>, typename _Pred = equal_to<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_↵`
`Allocator>>>`
unordered_multimap (`_InputIterator`, `_InputIterator`, `unordered_multimap< int, int >::size_type={}`, `_Hash=↵`
`Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_multimap< __iter_key_t< _InputIterator >, ↵`
`iter_val_t< _InputIterator >, _Hash, _Pred, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
unordered_multimap (`initializer_list< pair< _Key, _Tp > >`, `_Allocator`) -> `unordered_multimap< _Key, _Tp,`
`hash< _Key >, equal_to< _Key >, _Allocator >`

- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
unordered_multimap (`initializer_list`< `pair`< _Key, _Tp > >, `unordered_multimap`< int, int >::size_type, _Allocator) -> `unordered_multimap`< _Key, _Tp, `hash`< _Key >, `equal_to`< _Key >, _Allocator >
- `template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_multimap (`initializer_list`< `pair`< _Key, _Tp > >, `unordered_multimap`< int, int >::size_type, _Hash, _Allocator) -> `unordered_multimap`< _Key, _Tp, _Hash, `equal_to`< _Key >, _Allocator >
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_multimap (`initializer_list`< `pair`< _Key, _Tp > >, `unordered_multimap`< int, int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_multimap`< _Key, _Tp, _Hash, _Pred, _Allocator >
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (_InputIterator, _InputIterator, `unordered_multiset`< int >::size_type, _Allocator) -> `unordered_multiset`< typename `iterator_traits`< _InputIterator >::value_type, `hash`< typename `iterator_traits`< _InputIterator >::value_type >, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (_InputIterator, _InputIterator, `unordered_multiset`< int >::size_type, _Hash, _Allocator) -> `unordered_multiset`< typename `iterator_traits`< _InputIterator >::value_type, _Hash, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >
- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (_InputIterator, _InputIterator, `unordered_multiset`< int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_multiset`< typename `iterator_traits`< _InputIterator >::value_type, _Hash, _Pred, _Allocator >
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type, _Allocator) -> `unordered_multiset`< _Tp, `hash`< _Tp >, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type, _Hash, _Allocator) -> `unordered_multiset`< _Tp, _Hash, `equal_to`< _Tp >, _Allocator >
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
unordered_multiset (`initializer_list`< _Tp >, `unordered_multiset`< int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> `unordered_multiset`< _Tp, _Hash, _Pred, _Allocator >
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
unordered_set (_InputIterator, _InputIterator, `unordered_set`< int >::size_type, _Allocator) -> `unordered_set`< typename `iterator_traits`< _InputIterator >::value_type, `hash`< typename `iterator_traits`< _InputIterator >::value_type >, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
unordered_set (_InputIterator, _InputIterator, `unordered_set`< int >::size_type, _Hash, _Allocator) -> `unordered_set`< typename `iterator_traits`< _InputIterator >::value_type, _Hash, `equal_to`< typename `iterator_traits`< _InputIterator >::value_type >, _Allocator >

- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
`unordered_set` (`_InputIterator`, `_InputIterator`, `unordered_set< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_set< typename iterator_traits<_InputIterator>::value_type, _Hash, _Pred, _Allocator >`
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`unordered_set` (`initializer_list<_Tp>`, `unordered_set< int >::size_type`, `_Allocator`) -> `unordered_set< _Tp, hash<_Tp>, equal_to<_Tp>, _Allocator >`
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
`unordered_set` (`initializer_list<_Tp>`, `unordered_set< int >::size_type`, `_Hash`, `_Allocator`) -> `unordered_set< _Tp, _Hash, equal_to<_Tp>, _Allocator >`
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
`unordered_set` (`initializer_list<_Tp>`, `unordered_set< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_set< _Tp, _Hash, _Pred, _Allocator >`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`vector` (`_InputIterator`, `_InputIterator`, `_Allocator=_Allocator()`) -> `vector<_ValT, _Allocator >`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>>`
`vector` (`size_t`, `_Tp`, `_Allocator=_Allocator()`) -> `vector<_Tp, _Allocator >`

4.11.1 Detailed Description

GNU debug code, replaces standard behavior with debug behavior.

Macros and namespaces used by the implementation outside of debug wrappers to verify certain properties. The `__glibcxx_requires_xxx` macros are merely wrappers around the `__glibcxx_check_xxx` wrappers when we are compiling with debug mode, but disappear when we are in release mode so that there is no checking performed in, e.g., the standard library algorithms.

4.11.2 Function Documentation

operator<=()

```
template<typename _Tp, typename _Alloc >
bool std::__debug::operator<= (
    const forward_list<_Tp, _Alloc> & __lx,
    const forward_list<_Tp, _Alloc> & __ly) [inline]
```

Based on operator<.

operator>()

```
template<typename _Tp, typename _Alloc >
bool std::__debug::operator> (
    const forward_list<_Tp, _Alloc> & __lx,
    const forward_list<_Tp, _Alloc> & __ly) [inline]
```

Based on operator<.

operator>=()

```
template<typename _Tp, typename _Alloc >
bool std::__debug::operator>= (
```

```
const forward_list< _Tp, _Alloc > & __lx,
const forward_list< _Tp, _Alloc > & __ly ) [inline]
```

Based on operator<.

swap()

```
template<typename _Tp , typename _Alloc >
void std::__debug::swap (
    forward_list< _Tp, _Alloc > & __lx,
    forward_list< _Tp, _Alloc > & __ly ) [inline], [noexcept]
```

See std::forward_list::swap().

4.12 std::__detail Namespace Reference

Classes

- struct [_BracketMatcher](#)
- class [_Compiler](#)
- class [_Executor](#)
- struct [_List_node_base](#)
- struct [_List_node_header](#)
- struct [_Quoted_string](#)
- class [_Scanner](#)
- class [_StateSeq](#)

Typedefs

- using [__bare_timed_wait](#) = [__timed_waiter](#)< [std::false_type](#) >
- using [__bare_wait](#) = [__waiter](#)< [std::false_type](#) >
- template<typename _Cat, typename _Limit, typename _Otherwise = _Cat>
 using [__clamp_iter_cat](#) = [__conditional_t](#)< [derived_from](#)< _Cat, _Limit >, _Limit, _Otherwise >
- template<typename _Tp, typename _Up >
 using [__cmp3way_res_t](#) = [decltype](#)([std::declval](#)< _Tp >() <= [std::declval](#)< _Up >())
- template<typename _Tp >
 using [__cref](#) = const [remove_reference_t](#)< _Tp > &
- using [__enters_timed_wait](#) = [__timed_waiter](#)< [std::true_type](#) >
- using [__enters_wait](#) = [__waiter](#)< [std::true_type](#) >
- template<typename _Tp >
 using [__integer_from_chars_result_type](#) = [enable_if_t](#)< [__or_](#)< [__is_signed_integer](#)< _Tp >, [__is_](#)↔
[unsigned_integer](#)< _Tp >, [is_same](#)< char, [remove_cv_t](#)< _Tp > > >::value, [from_chars_result](#) >
- template<typename _Tp >
 using [__integer_to_chars_result_type](#) = [enable_if_t](#)< [__or_](#)< [__is_signed_integer](#)< _Tp >, [__is_](#)↔
[integer](#)< _Tp >, [is_same](#)< char, [remove_cv_t](#)< _Tp > > >::value, [to_chars_result](#) >
- template<typename _Iter >
 using [__iter_concept](#) = typename [__iter_concept_impl](#)< _Iter >::type
- template<typename _Tp >
 using [__iter_diff_t](#) = typename [__iter_traits](#)< _Tp, [incrementable_traits](#)< _Tp > >::difference_type
- template<typename _Iter, typename _Tp = _Iter>
 using [__iter_traits](#) = typename [__iter_traits_impl](#)< _Iter, _Tp >::type
- template<typename _Tp >
 using [__iter_value_t](#) = typename [__iter_traits](#)< _Tp, [indirectly_readable_traits](#)< _Tp > >::value_type
- using [__platform_wait_t](#) = [uint64_t](#)
- template<typename _Tp >
 using [__range_iter_t](#) = [decltype](#)([ranges::__cust_access::__begin](#)([std::declval](#)< _Tp & >()))

- template<typename _Tp, typename _Up = _Tp>
using **__synth3way_t** = decltype(__detail::__synth3way(std::declval<_Tp &>(), std::declval<_Up &>()))
- template<typename _Tp >
using **__unsigned_least_t** = typename __to_chars_unsigned_type<_Tp >::type
- using **__wait_clock_t** = [chrono::steady_clock](#)
- template<typename _Tp >
using **__with_ref** = _Tp &
- template<typename _CharT >
using **_Matcher** = std::function< bool(_CharT)>
- typedef long **_StateldT**

Enumerations

- enum [_Opcode](#) : int {
 _S_opcode_unknown, **_S_opcode_alternative**, **_S_opcode_repeat**, **_S_opcode_backref**,
 _S_opcode_line_begin_assertion, **_S_opcode_line_end_assertion**, **_S_opcode_word_boundary**, **_S_opcode_subexpr_lookahead**,
 _S_opcode_subexpr_begin, **_S_opcode_subexpr_end**, **_S_opcode_dummy**, **_S_opcode_match**,
 _S_opcode_accept }
• enum class **_RegexExecutorPolicy** : int { **_S_auto**, **_S_alternate** }

Functions

- template<typename _Res, typename _Tp >
constexpr _Res **__abs_r** (_Tp __val)
- template<typename >
void **__abs_r** (bool)=delete
- template<typename _Tp >
bool **__atomic_compare** (const _Tp &__a, const _Tp &__b)
- template<typename _Pred, typename _Spin = __default_spin_policy>
bool **__atomic_spin** (_Pred &__pred, _Spin __spin=_Spin{ }) noexcept
- template<typename... _Ts>
constexpr auto **__common_cmp_cat** ()
- template<typename _Clock, typename _Dur >
bool **__cond_wait_until** (__condvar &__cv, [mutex](#) &__mx, const [chrono::time_point](#)< _Clock, _Dur > &__atime)
- template<typename _Clock, typename _Dur >
bool **__cond_wait_until_impl** (__condvar &__cv, [mutex](#) &__mx, const [chrono::time_point](#)< _Clock, _Dur > &__atime)
- template<typename _Container, typename _UnsafeContainer, typename _Predicate >
_Container::size_type **__erase_nodes_if** (_Container &__cont, _UnsafeContainer &__ucont, _Predicate __pred)
- template<typename _ValT, typename _CharT, typename _Traits >
[basic_istream](#)< _CharT, _Traits > & **__extract_params** ([basic_istream](#)< _CharT, _Traits > &__is, [vector](#)< _ValT > &__vals, size_t __n)
- template<bool _DecOnly, typename _Tp >
bool **__from_chars_alnum** (const char *&__first, const char *__last, _Tp &__val, int __base)
- template<bool _DecOnly = false>
unsigned char **__from_chars_alnum_to_val** (unsigned char __c)
- constexpr auto **__from_chars_alnum_to_val_table** ()
- template<bool _DecOnly, typename _Tp >
bool **__from_chars_pow2_base** (const char *&__first, const char *__last, _Tp &__val, int __base)
- template<typename _Tp >
constexpr _Tp **__gcd** (_Tp __m, _Tp __n)

- `template<typename _Tp >`
`constexpr bool __p1_representable_as_double (_Tp __x) noexcept`
- `template<typename _Tp >`
`bool __raise_and_add (_Tp &__val, int __base, unsigned char __c)`
- `template<typename _Biliter, typename _Alloc, typename _CharT, typename _TraitsT >`
`bool __regex_algo_impl (_Biliter __s, _Biliter __e, match_results< _Biliter, _Alloc > &__m, const basic_regex< _CharT, _TraitsT > &__re, regex_constants::match_flag_type __flags, _RegexExecutorPolicy __policy, bool __↵
match_mode)`
- `template<typename _Tp >`
`constexpr bool __representable_as_double (_Tp __x) noexcept`
- `template<typename _Tp >`
`void __return_temporary_buffer (_Tp *__p, size_t __len)`
- `void __thread_relax () noexcept`
- `void __thread_yield () noexcept`
- `template<typename _Tp >`
`to_chars_result __to_chars (char *__first, char *__last, _Tp __val, int __base) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type< _Tp > __to_chars_10 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`void __to_chars_10_impl (char *__first, unsigned __len, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type< _Tp > __to_chars_16 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type< _Tp > __to_chars_2 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type< _Tp > __to_chars_8 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`constexpr unsigned __to_chars_len (_Tp __value, int __base) noexcept`
- `template<typename _Tp >`
`constexpr unsigned __to_chars_len_2 (_Tp __value) noexcept`
- `template<typename _Dur >`
`__wait_clock_t::time_point __to_wait_clock (const chrono::time_point< __wait_clock_t, _Dur > &__atime)
noexcept`
- `template<typename _Clock, typename _Dur >`
`__wait_clock_t::time_point __to_wait_clock (const chrono::time_point< _Clock, _Dur > &__atime) noexcept`
- `template<typename _Tp >`
`constexpr bool __Power_of_2 (_Tp __x)`
- `template<typename _CharT, typename _Traits, typename _String >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
__Quoted_string< _String, _CharT > &__str)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
__Quoted_string< const _CharT *, _CharT > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, const
__Quoted_string< basic_string< _CharT, _Traits, _Alloc > &, _CharT > &__str)`

Variables

- `constexpr auto __atomic_spin_count`
- `constexpr auto __atomic_spin_count_relax`
- `template<typename _Tp >`
`constexpr unsigned __cmp_cat_id`

- `template<> constexpr unsigned __cmp_cat_id< partial_ordering >`
- `template<> constexpr unsigned __cmp_cat_id< strong_ordering >`
- `template<> constexpr unsigned __cmp_cat_id< weak_ordering >`
- `template<typename _Tp >`
`constexpr bool __is_span`
- `template<typename _Tp, size_t _Num>`
`constexpr bool __is_span< span< _Tp, _Num > >`
- `template<typename _Tp >`
`constexpr bool __is_std_array`
- `constexpr size_t __platform_wait_alignment`
- `constexpr struct std::__detail::__Synth3way __synth3way`
- `constexpr _StateIdT __S_invalid_state_id`

4.12.1 Detailed Description

Implementation details not part of the namespace std interface.

4.12.2 Function Documentation

`__from_chars_alnum()`

```
template<bool _DecOnly, typename _Tp >
bool std::__detail::__from_chars_alnum (
    const char *& __first,
    const char * __last,
    _Tp & __val,
    int __base )
```

std::from_chars implementation for integers in any base. If _DecOnly is true, then we may assume __base is at most 10.

`__from_chars_pow2_base()`

```
template<bool _DecOnly, typename _Tp >
bool std::__detail::__from_chars_pow2_base (
    const char *& __first,
    const char * __last,
    _Tp & __val,
    int __base )
```

std::from_chars implementation for integers in a power-of-two base. If _DecOnly is true, then we may assume __base is at most 8.

`operator<<() [1/2]`

```
template<typename _CharT, typename _Traits, typename _String >
std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const _Quoted_string< _String, _CharT > & __str )
```

Insertion for quoted strings.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 2344 `quoted()`'s interaction with padding is unclear

`operator<<() [2/2]`

```
template<typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< (
```



```
std::basic_ostream< _CharT, _Traits > & __os,
const _Quoted_string< const _CharT *, _CharT > & __str )
```

Insertor for quoted strings.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 2344 `quoted()`'s interaction with padding is unclear

operator>>()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_istream< _CharT, _Traits > & std::__detail::operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    const _Quoted_string< basic_string< _CharT, _Traits, _Alloc > &, _CharT > & __str )
```

Extractor for delimited strings. The left and right delimiters can be different.

References `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::flags()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::ios_base::setf()`, `std::ios_base::skipws`, and `std::basic_istream< _CharT, _Traits >::unget()`.

4.13 std::__parallel Namespace Reference

Classes

- struct `_CRandNumber`

Functions

- template<typename __RAIter, typename _Tp, typename _BinaryOperation >
_Tp **accumulate_switch** (__RAIter __begin, __RAIter __end, _Tp __init, _BinaryOperation __binary_op, `random_access_iterator_tag`, `__gnu_parallel::Parallelism` __parallelism_tag)
- template<typename _Iter, typename _Tp, typename _BinaryOperation, typename _IteratorTag >
_Tp **accumulate_switch** (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, _IteratorTag)
- template<typename _Iter, typename _Tp, typename _IteratorTag >
_Tp **accumulate_switch** (_Iter __begin, _Iter __end, _Tp __init, _IteratorTag)
- template<typename _Iter, typename _Tp, typename _BinaryOper, typename _Tag >
_Tp **accumulate_switch** (_Iter, _Iter, _Tp, _BinaryOper, _Tag)
- template<typename _Iter, typename _Tp, typename _Tag >
_Tp **accumulate_switch** (_Iter, _Iter, _Tp, _Tag)
- template<typename _RAIter, typename _Tp, typename _BinaryOper >
_Tp **accumulate_switch** (_RAIter, _RAIter, _Tp, _BinaryOper, `random_access_iterator_tag`, `__gnu_parallel::Parallelism` __parallelism=`__gnu_parallel::parallel_unbalanced`)
- template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >
_OutputIterator **adjacent_difference_switch** (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, _IteratorTag1, _IteratorTag2)
- template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >
_OutputIterator **adjacent_difference_switch** (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, `random_access_iterator_tag`, `random_access_iterator_tag`, `__gnu_parallel::Parallelism` __parallelism_tag)
- template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >
_OIter **adjacent_difference_switch** (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)
- template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter **adjacent_difference_switch** (_Iter, _Iter, _OIter, _BinaryOper, `random_access_iterator_tag`, `random_access_iterator_tag`, `__gnu_parallel::Parallelism` __parallelism=`__gnu_parallel::parallel_unbalanced`)
- template<typename _Filter, typename _BiPredicate, typename _IterTag >
_Filter **adjacent_find_switch** (_Filter, _Filter, _BiPredicate, _IterTag)
- template<typename _Filter, typename _IterTag >
_Filter **adjacent_find_switch** (_Filter, _Filter, _IterTag)

- `template<typename _Iterator, typename _BinaryPredicate, typename _IteratorTag >`
`_Iterator __adjacent_find_switch (_Iterator __begin, _Iterator __end, _BinaryPredicate __pred, _Iterator↵`
`Tag)`
- `template<typename _Iterator, typename _IteratorTag >`
`_Iterator __adjacent_find_switch (_Iterator __begin, _Iterator __end, _IteratorTag)`
- `template<typename _RAIter, typename _BinaryPredicate >`
`_RAIter __adjacent_find_switch (_RAIter __begin, _RAIter __end, _BinaryPredicate __pred, random_access_iterator_tag)`
- `template<typename _RAIter >`
`_RAIter __adjacent_find_switch (_RAIter __begin, _RAIter __end, random_access_iterator_tag)`
- `template<typename _RAIter, typename _BiPredicate >`
`_RAIter __adjacent_find_switch (_RAIter, _RAIter, _BiPredicate, random_access_iterator_tag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type __count_if_switch (_Iter __begin, _Iter __end, _Predicate __pred,`
`_IteratorTag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`iterator_traits< _Iter >::difference_type __count_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename _RAIter, typename _Predicate >`
`iterator_traits< _RAIter >::difference_type __count_if_switch (_RAIter __begin, _RAIter __end, _Predicate ↵`
`__pred, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type __count_switch (_Iter __begin, _Iter __end, const _Tp &__value, ↵`
`_IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`iterator_traits< _Iter >::difference_type __count_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`iterator_traits< _RAIter >::difference_type __count_switch (_RAIter __begin, _RAIter __end, const _Tp &↵`
`__value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool __equal_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred,`
`_IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool __equal_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate`
`__pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _Filterator, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter __find_first_of_switch (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2, _Binary↵`
`Predicate __comp, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _Filterator, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter __find_first_of_switch (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2, _Iterator↵`
`Tag1, _IteratorTag2)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Iter __find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _Filter, typename _IterTag1, typename _IterTag2 >`
`_Iter __find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _Filterator, typename _BinaryPredicate, typename _IteratorTag >`
`_RAIter __find_first_of_switch (_RAIter __begin1, _RAIter __end1, _Filterator __begin2, _Filterator __end2,`
`_BinaryPredicate __comp, random_access_iterator_tag, _IteratorTag)`
- `template<typename _RAIter, typename _Filter, typename _BiPredicate, typename _IterTag >`
`_RAIter __find_first_of_switch (_RAIter, _RAIter, _Filter, _Filter, _BiPredicate, random_access_iterator_tag, ↵`
`_IterTag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`_Iter __find_if_switch (_Iter __begin, _Iter __end, _Predicate __pred, _IteratorTag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`_Iter __find_if_switch (_Iter, _Iter, _Predicate, _IterTag)`

- `template<typename _RAIter, typename _Predicate >`
`_RAIter __find_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Iter __find_switch (_Iter __begin, _Iter __end, const _Tp &__val, _IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`_Iter __find_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`_RAIter __find_switch (_RAIter __begin, _RAIter __end, const _Tp &__val, random_access_iterator_tag)`
- `template<typename _Iter, typename _Function, typename _IteratorTag >`
`_Function __for_each_switch (_Iter __begin, _Iter __end, _Function __f, _IteratorTag)`
- `template<typename _Iter, typename _Function, typename _IterTag >`
`_Function __for_each_switch (_Iter, _Iter, _Function, _IterTag)`
- `template<typename _RAIter, typename _Function >`
`_Function __for_each_switch (_RAIter __begin, _RAIter __end, _Function __f, random_access_iterator_tag,
gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _OIter, typename _Size, typename _Generator, typename _IterTag >`
`_OIter __generate_n_switch (_OIter, _Size, _Generator, _IterTag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator, typename _IteratorTag >`
`_OutputIterator __generate_n_switch (_OutputIterator __begin, _Size __n, _Generator __gen, _IteratorTag)`
- `template<typename _RAIter, typename _Size, typename _Generator >`
`_RAIter __generate_n_switch (_RAIter __begin, _Size __n, _Generator __gen, random_access_iterator_tag,
gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Generator, typename _IterTag >`
`void __generate_switch (_Filter, _Filter, _Generator, _IterTag)`
- `template<typename _Filterator, typename _Generator, typename _IteratorTag >`
`void __generate_switch (_Filterator __begin, _Filterator __end, _Generator __gen, _IteratorTag)`
- `template<typename _RAIter, typename _Generator >`
`void __generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_access_iterator_tag,
gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _IteratorTag1, typename _IteratorTag2 >`
`_Tp __inner_product_switch (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _Tag1, typename _Tag2 >`
`_Tp __inner_product_switch (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, _Tag1, _Tag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp __inner_product_switch (_RAIter1, _RAIter1, _RAIter2, _Tp, _BinaryFunction1, _BinaryFunction2, random_access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism = gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool __lexicographical_compare_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`bool __lexicographical_compare_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool __lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter __max_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator __max_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IteratorTag)`

- `template<typename _RAIter, typename _Compare >`
`_RAIter __max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag,
__gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator __merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, random_access_iterator_tag, random_access_iterator_tag,
random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter __merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter __min_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator __min_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IteratorTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter __min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag,
__gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2,
_Predicate __pred, _IteratorTag1, _IteratorTag2\)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred,
_IteratorTag1, _IteratorTag2\)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1, _Iter1, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > __mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2,
_Predicate __pred, random_access_iterator_tag, random_access_iterator_tag\)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > __mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2,
_RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag\)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator __partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation
__bin_op, _IteratorTag1, _IteratorTag2\)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation
__bin_op, random_access_iterator_tag, random_access_iterator_tag\)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >`
`_OIter __partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter __partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _IterTag >`
`_Filter __partition_switch (_Filter, _Filter, _Predicate, _IterTag)`
- `template<typename _Filterator, typename _Predicate, typename _IteratorTag >`
`_Filterator __partition_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, _IteratorTag)`

- `template<typename _RAIter, typename _Predicate >`
`_RAIter __partition_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp, typename _IterTag >`
`void __replace_if_switch (_Filter, _Filter, _Predicate, const _Tp &, _IterTag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp, typename _IteratorTag >`
`void __replace_if_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp & __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate, typename _Tp >`
`void __replace_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, const _Tp & __new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Tp, typename _IterTag >`
`void __replace_switch (_Filter, _Filter, const _Tp &, const _Tp &, _IterTag)`
- `template<typename _Filterator, typename _Tp, typename _IteratorTag >`
`void __replace_switch (_Filterator __begin, _Filterator __end, const _Tp & __old_value, const _Tp & __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`void __replace_switch (_RAIter __begin, _RAIter __end, const _Tp & __old_value, const _Tp & __new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate, typename _IterTag >`
`_Filter __search_n_switch (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, _IterTag)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator __search_n_switch (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred, _IteratorTag)`
- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_RAIter __search_n_switch (_RAIter __begin, _RAIter __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred, random_access_iterator_tag)`
- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_RAIter __search_n_switch (_RAIter, _RAIter, _Integer, const _Tp &, _BiPredicate, random_access_iterator_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Filter1 __search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Filter1, typename _Filter2, typename _IterTag1, typename _IterTag2 >`
`_Filter1 __search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _IterTag1, _IterTag2)`
- `template<typename _Filterator1, typename _Filterator2, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Filterator1 __search_switch (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _BinaryPredicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _Filterator1, typename _Filterator2, typename _IteratorTag1, typename _IteratorTag2 >`
`_Filterator1 __search_switch (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _BinaryPredicate >`
`_RAIter1 __search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _BinaryPredicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2 >`
`_RAIter1 __search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _BiPredicate >`
`_RAIter1 __search_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter2, _BiPredicate, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __set_difference_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`

- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __set_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter __set_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __set_intersection_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __set_intersection_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter __set_intersection_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __set_symmetric_difference_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __set_symmetric_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter __set_symmetric_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __set_union_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __set_union_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter __set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation, typename _IterTag1, typename _IterTag2 >`
`_OIter __transform1_switch (_Iter, _Iter, _OIter, _UnaryOperation, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _RAOIter, typename _UnaryOperation >`
`_RAOIter __transform1_switch (_RAIter, _RAIter, _RAOIter, _UnaryOperation, random_access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism=gnu_parallel::parallel_balanced)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_RAIter2 __transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation __unary_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation >`
`_RAIter2 __transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation`

```

__unary_op, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __↵
parallelism_tag)
• template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation, typename _Tag1, typename _Tag2
, typename _Tag3 >
  _OutputIterator transform2_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __↵
result, _BinaryOperation __binary_op, _Tag1, _Tag2, _Tag3)
• template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation, typename _Tag1, typename _Tag2, typename
_↵
Tag3 >
  _OIter transform2_switch (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, _Tag1, _Tag2, _Tag3)
• template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BinaryOperation >
  _RAIter3 transform2_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter3 __result, ↵
_BinaryOperation __binary_op, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag,
__gnu_parallel::Parallelism __parallelism_tag)
• template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BiOperation >
  _RAIter3 transform2_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter3, _BiOperation, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism= __gnu_parallel::parallel_ba
• template<typename _Iter, typename _OutputIterator, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >
  _OutputIterator unique_copy_switch (_Iter __begin, _Iter __last, _OutputIterator __out, _Predicate __pred,
_↵
IteratorTag1, _IteratorTag2)
• template<typename _Iter, typename _OIter, typename _Predicate, typename _IterTag1, typename _IterTag2 >
  _OIter unique_copy_switch (_Iter, _Iter, _OIter, _Predicate, _IterTag1, _IterTag2)
• template<typename _RAIter, typename _RandomAccessOutputIterator, typename _Predicate >
  _RandomAccessOutputIterator unique_copy_switch (_RAIter __begin, _RAIter __last, _RandomAccess↵
OutputIterator __out, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)
• template<typename _RAIter, typename _RandomAccess_OIter, typename _Predicate >
  _RandomAccess_OIter unique_copy_switch (_RAIter, _RAIter, _RandomAccess_OIter, _Predicate,
random_access_iterator_tag, random_access_iterator_tag)
• template<typename _Iter, typename _Tp, typename _BinaryOperation >
  _Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op)
• template<typename _Iter, typename _Tp, typename _BinaryOperation >
  _Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::Parallelism
__parallelism_tag)
• template<typename _Iter, typename _Tp, typename _BinaryOperation >
  _Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::sequential_tag)
• template<typename _Iter, typename _Tp >
  _Tp accumulate (_Iter, _Iter, _Tp)
• template<typename _Iter, typename _Tp >
  _Tp accumulate (_Iter, _Iter, _Tp, __gnu_parallel::Parallelism)
• template<typename _Iter, typename _Tp >
  _Tp accumulate (_Iter, _Iter, _Tp, __gnu_parallel::sequential_tag)
• template<typename _Iter, typename _Tp, typename _BinaryOper >
  _Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper)
• template<typename _Iter, typename _Tp, typename _BinaryOper >
  _Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::Parallelism)
• template<typename _Iter, typename _Tp, typename _BinaryOper >
  _Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::sequential_tag)
• template<typename _Iter, typename _OutputIterator >
  _OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result)
• template<typename _Iter, typename _OutputIterator >
  _OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::Parallelism
__parallelism_tag)
• template<typename _Iter, typename _OutputIterator >
  _OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)

```


- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >
_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >
_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >
_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OIter >
_OIter adjacent_difference (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter >
_OIter adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter >
_OIter adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >
_Filter adjacent_find (_Filter, _Filter)`
- `template<typename _Filter >
_Filter adjacent_find (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _BiPredicate >
_Filter adjacent_find (_Filter, _Filter, _BiPredicate)`
- `template<typename _Filter, typename _BiPredicate >
_Filter adjacent_find (_Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator >
_FIterator adjacent_find (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator >
_FIterator adjacent_find (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _BinaryPredicate >
_FIterator adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _BinaryPredicate >
_FIterator adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __pred)`
- `template<typename _Iter, typename _Tp >
iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value)`
- `template<typename _Iter, typename _Tp >
iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp >
iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >
iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Predicate >
iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Predicate >
iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`constexpr bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __↵`
`binary_pred)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`constexpr bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Iter find (_Iter __begin, _Iter __end, const _Tp &__val)`
- `template<typename _Iter, typename _Tp >`
`_Iter find (_Iter __begin, _Iter __end, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _FIterator >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2)`
- `template<typename _Iter, typename _FIterator >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _BinaryPredicate`
`__comp)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _BinaryPredicate`
`__comp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter)`
- `template<typename _Iter, typename _Filter >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Function >`
`_Function for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Function >`
`_Function for_each (_Iter, _Iter, _Function)`
- `template<typename _Iterator, typename _Function >`
`_Function for_each (_Iterator __begin, _Iterator __end, _Function __f)`
- `template<typename _Iterator, typename _Function >`
`_Function for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::Parallelism __↵`
`parallelism_tag)`

- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator)`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator, __gnu_parallel::__Parallelism)`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen, __gnu_parallel::__Parallelism __←
parallelism_tag)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen, __gnu_parallel::sequential_tag)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator, __gnu_parallel::__Parallelism)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu_parallel::__Parallelism
__parallelism_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Tp >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp)`
- `template<typename _IIter1, typename _IIter2, typename _Tp >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp, __gnu_parallel::__Parallelism)`
- `template<typename _IIter1, typename _IIter2, typename _Tp >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp, _BinaryFunction1, _BinaryFunction2)`
- `template<typename _IIter1, typename _IIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::__Parallelism)`
- `template<typename _IIter1, typename _IIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_IIter1, _IIter1, _IIter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2 >`
`constexpr bool lexicographical_compare (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2)`
- `template<typename _IIter1, typename _IIter2 >`
`bool lexicographical_compare (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2,
__gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate >`
`constexpr bool lexicographical_compare (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2,
_Predicate __pred)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate >`
`bool lexicographical_compare (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _Predicate
__pred, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter, __gnu_parallel::__Parallelism)`

- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator >`
`_FIterator max_element (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, _Compare __comp)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵
result)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵
result, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵
result, _Compare __comp)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵
result, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Compare >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Compare)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Compare >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`

- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __begin1, _InputIterator1 __end1, _InputIterator2 __begin2, _InputIterator2 __end2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end)`
- `template<typename _RAIter >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end)`
- `template<typename _RAIter >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary↵`
`__op)`
- `template<typename _Iter, typename _OIter >`
`_OIter partial_sum (_Iter, _Iter, _OIter __result)`
- `template<typename _Iter, typename _OIter >`
`_OIter partial_sum (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter partial_sum (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter partial_sum (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate >`
`_Filter partition (_Filter, _Filter, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`_Filter partition (_Filter, _Filter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator partition (_Filterator __begin, _Filterator __end, _Predicate __pred)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator partition (_Filterator __begin, _Filterator __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter >`
`void random_shuffle (_RAlter __begin, _RAlter __end)`
- `template<typename _RAlter >`
`void random_shuffle (_RAlter __begin, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &__rand, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value,`
`__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value)`

- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value,`
`__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator1, typename _Filterator2 >`
`_Filterator1 search (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2)`
- `template<typename _Filterator1, typename _Filterator2 >`
`_Filterator1 search (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Filterator1, typename _Filterator2, typename _BinaryPredicate >`
`_Filterator1 search (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _↵`
`BinaryPredicate __pred)`
- `template<typename _Filterator1, typename _Filterator2, typename _BinaryPredicate >`
`_Filterator1 search (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _↵`
`BinaryPredicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _ForwardIterator, typename _Searcher >`
`_ForwardIterator search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &__searcher)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Integer, typename _Tp >`
`_Filterator search_n (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp &__val)`
- `template<typename _Filterator, typename _Integer, typename _Tp >`
`_Filterator search_n (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_Filterator search_n (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp &__val, _BinaryPredicate`
`__binary_pred)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_Filterator search_n (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp &__val, _BinaryPredicate`
`__binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iiter1, typename _Iiter2, typename _OutputIterator >`
`_OutputIterator set_difference (_Iiter1 __begin1, _Iiter1 __end1, _Iiter2 __begin2, _Iiter2 __end2, _Output↵`
`Iiterator __out)`
- `template<typename _Iiter1, typename _Iiter2, typename _OutputIterator >`
`_OutputIterator set_difference (_Iiter1 __begin1, _Iiter1 __end1, _Iiter2 __begin2, _Iiter2 __end2, _Output↵`
`Iiterator __out, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_exact_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_sampling_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`

- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::BinaryOperation __binary_op)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::BinaryOperation __binary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`

- `template<typename _Iter , typename _OIter >
_OIter unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter , typename _OIter >
_OIter unique_copy (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter , typename _OIter , typename _Predicate >
_OIter unique_copy (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Iter , typename _OIter , typename _Predicate >
_OIter unique_copy (_Iter, _Iter, _OIter, _Predicate, __gnu_parallel::sequential_tag)`

4.13.1 Detailed Description

GNU parallel code, replaces standard behavior with parallel behavior.

4.13.2 Function Documentation

search()

```
template<typename _ForwardIterator , typename _Searcher >
_FowardIterator std::__parallel::search (
    _ForwardIterator __first,
    _ForwardIterator __last,
    const _Searcher & __searcher ) [inline]
```

Search a sequence using a Searcher object.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__searcher</code>	A callable object.

Returns

`__searcher(__first, __last).first`

4.14 std::chrono Namespace Reference

Classes

- class [duration](#)
- struct [duration_values](#)
- struct [steady_clock](#)
- struct [system_clock](#)
- class [time_point](#)
- struct [treat_as_floating_point](#)

Typedefs

- using `days` = [duration](#)< `int64_t`, [ratio](#)< 86400 > >
- using `file_clock` = ::std::filesystem::__file_clock
- `template<typename _Duration >`
using `file_time` = [time_point](#)< `file_clock`, `_Duration` >
- using `gps_seconds` = [gps_time](#)< [seconds](#) >
- `template<typename _Duration >`
using `gps_time` = [time_point](#)< `gps_clock`, `_Duration` >

- using `high_resolution_clock` = `system_clock`
- using `hours` = `duration`< `int64_t`, `ratio`< 3600 > >
- using `local_days` = `local_time`< `days` >
- using `local_seconds` = `local_time`< `seconds` >
- template<typename `_Duration` >
using `local_time` = `time_point`< `local_t`, `_Duration` >
- using `microseconds` = `duration`< `int64_t`, `micro` >
- using `milliseconds` = `duration`< `int64_t`, `milli` >
- using `minutes` = `duration`< `int64_t`, `ratio`< 60 > >
- using `months` = `duration`< `int64_t`, `ratio`< 2629746 > >
- using `nanoseconds` = `duration`< `int64_t`, `nano` >
- using `seconds` = `duration`< `int64_t` >
- using `sys_days` = `sys_time`< `days` >
- using `sys_seconds` = `sys_time`< `seconds` >
- template<typename `_Duration` >
using `sys_time` = `time_point`< `system_clock`, `_Duration` >
- using `tai_seconds` = `tai_time`< `seconds` >
- template<typename `_Duration` >
using `tai_time` = `time_point`< `tai_clock`, `_Duration` >
- using `utc_seconds` = `utc_time`< `seconds` >
- template<typename `_Duration` >
using `utc_time` = `time_point`< `utc_clock`, `_Duration` >
- using `weeks` = `duration`< `int64_t`, `ratio`< 604800 > >
- using `years` = `duration`< `int64_t`, `ratio`< 31556952 > >

Functions

- template<typename `_Rep`, typename `_Period` >
constexpr `enable_if_t`< `numeric_limits`< `_Rep` >::is_signed, `duration`< `_Rep`, `_Period` > > `abs` (`duration`< `_Rep`, `_Period` > `__d`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration`< `_ToDur` > `ceil` (const `duration`< `_Rep`, `_Period` > &`__d`)
- template<typename `_ToDur`, typename `_Clock`, typename `_Dur` >
constexpr `enable_if_t`< `__is_duration`< `_ToDur` >::value, `time_point`< `_Clock`, `_ToDur` > > `ceil` (const `time_point`< `_Clock`, `_Dur` > &`__tp`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration`< `_ToDur` > `duration_cast` (const `duration`< `_Rep`, `_Period` > &`__d`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `__enable_if_is_duration`< `_ToDur` > `floor` (const `duration`< `_Rep`, `_Period` > &`__d`)
- template<typename `_ToDur`, typename `_Clock`, typename `_Dur` >
constexpr `enable_if_t`< `__is_duration`< `_ToDur` >::value, `time_point`< `_Clock`, `_ToDur` > > `floor` (const `time_point`< `_Clock`, `_Dur` > &`__tp`)
- constexpr bool `is_am` (const `hours` &`__h`) noexcept
- constexpr bool `is_pm` (const `hours` &`__h`) noexcept
- constexpr `hours` `make12` (const `hours` &`__h`) noexcept
- constexpr `hours` `make24` (const `hours` &`__h`, bool `__is_pm`) noexcept
- template<typename `_CharT`, typename `_Traits`, typename `_Rep`, typename `_Period` >
`basic_ostream`< `_CharT`, `_Traits` > & `operator`<< (`std::basic_ostream`< `_CharT`, `_Traits` > &`__os`, const `duration`< `_Rep`, `_Period` > &`__d`)
- template<typename `_ToDur`, typename `_Rep`, typename `_Period` >
constexpr `enable_if_t`< `__and`< `__is_duration`< `_ToDur` >, `__not`< `treat_as_floating_point`< typename `_ToDur`::rep > >::value, `_ToDur` > `round` (const `duration`< `_Rep`, `_Period` > &`__d`)

- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename _ToDur::rep > > >::value, time_point< _Clock, _ToDur > > round (const time_point< _Clock, _Dur > &__tp)`
- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr __enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > time_point_cast (const time_point< _Clock, _Dur > &__t)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type operator- (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Rep2, typename _Period >`
`constexpr duration< __common_rep_t< _Rep2, _Rep1 >, _Period > operator* (const _Rep1 &__s, const duration< _Rep2, _Period > &__d)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > operator/ (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< _Rep1, _Rep2 >::type operator/ (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > operator% (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type operator% (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator< (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator!= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator<= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator> (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator>= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Clock, typename _Dur2 >`
`constexpr time_point< _Clock, typename common_type< duration< _Rep1, _Period1 >, _Dur2 >::type > operator+ (const duration< _Rep1, _Period1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >`
`constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type > operator- (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`

- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr common_type< _Dur1, _Dur2 >::type operator- (const time_point< _Clock, _Dur1 > &__lhs, const`
`time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator!= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 >`
`&__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator< (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 >`
`&__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator<= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 >`
`&__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator> (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 >`
`&__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator>= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 >`
`&__rhs)`

Variables

- `constexpr month April`
- `constexpr month August`
- `constexpr month December`
- `constexpr month February`
- `constexpr weekday Friday`
- `template<typename _Tp >`
`constexpr bool is_clock_v`
- `template<> constexpr bool is_clock_v< file_clock >`
- `template<> constexpr bool is_clock_v< gps_clock >`
- `template<> constexpr bool is_clock_v< steady_clock >`
- `template<> constexpr bool is_clock_v< system_clock >`
- `template<> constexpr bool is_clock_v< tai_clock >`
- `template<> constexpr bool is_clock_v< utc_clock >`
- `constexpr month January`
- `constexpr month July`
- `constexpr month June`
- `constexpr last_spec last`
- `constexpr month March`
- `constexpr month May`
- `constexpr weekday Monday`
- `constexpr month November`
- `constexpr month October`
- `constexpr weekday Saturday`
- `constexpr month September`
- `constexpr weekday Sunday`
- `constexpr weekday Thursday`
- `template<typename _Rep >`
`constexpr bool treat_as_floating_point_v`
- `constexpr weekday Tuesday`
- `constexpr weekday Wednesday`

4.14.1 Detailed Description

ISO C++ 2011 namespace for date and time utilities.

4.15 std::decimal Namespace Reference

Classes

- class [decimal128](#)
- class [decimal32](#)
- class [decimal64](#)

Functions

- double **decimal128_to_double** ([decimal128](#) __d)
- float **decimal128_to_float** ([decimal128](#) __d)
- long double **decimal128_to_long_double** ([decimal128](#) __d)
- long long **decimal128_to_long_long** ([decimal128](#) __d)
- double **decimal32_to_double** ([decimal32](#) __d)
- float **decimal32_to_float** ([decimal32](#) __d)
- long double **decimal32_to_long_double** ([decimal32](#) __d)
- long long **decimal32_to_long_long** ([decimal32](#) __d)
- double **decimal64_to_double** ([decimal64](#) __d)
- float **decimal64_to_float** ([decimal64](#) __d)
- long double **decimal64_to_long_double** ([decimal64](#) __d)
- long long **decimal64_to_long_long** ([decimal64](#) __d)
- double **decimal_to_double** ([decimal128](#) __d)
- double **decimal_to_double** ([decimal32](#) __d)
- double **decimal_to_double** ([decimal64](#) __d)
- float **decimal_to_float** ([decimal128](#) __d)
- float **decimal_to_float** ([decimal32](#) __d)
- float **decimal_to_float** ([decimal64](#) __d)
- long double **decimal_to_long_double** ([decimal128](#) __d)
- long double **decimal_to_long_double** ([decimal32](#) __d)
- long double **decimal_to_long_double** ([decimal64](#) __d)
- long long **decimal_to_long_long** ([decimal128](#) __d)
- long long **decimal_to_long_long** ([decimal32](#) __d)
- long long **decimal_to_long_long** ([decimal64](#) __d)
- static [decimal128](#) **make_decimal128** (long long __coeff, int __exp)
- static [decimal128](#) **make_decimal128** (unsigned long long __coeff, int __exp)
- static [decimal32](#) **make_decimal32** (long long __coeff, int __exp)
- static [decimal32](#) **make_decimal32** (unsigned long long __coeff, int __exp)
- static [decimal64](#) **make_decimal64** (long long __coeff, int __exp)
- static [decimal64](#) **make_decimal64** (unsigned long long __coeff, int __exp)
- bool **operator!=** ([decimal128](#) __lhs, [decimal128](#) __rhs)
- bool **operator!=** ([decimal128](#) __lhs, [decimal32](#) __rhs)
- bool **operator!=** ([decimal128](#) __lhs, [decimal64](#) __rhs)
- bool **operator!=** ([decimal128](#) __lhs, int __rhs)
- bool **operator!=** ([decimal128](#) __lhs, long __rhs)
- bool **operator!=** ([decimal128](#) __lhs, long long __rhs)
- bool **operator!=** ([decimal128](#) __lhs, unsigned int __rhs)
- bool **operator!=** ([decimal128](#) __lhs, unsigned long __rhs)

- bool **operator!=** (decimal128 __lhs, unsigned long long __rhs)
- bool **operator!=** (decimal32 __lhs, decimal128 __rhs)
- bool **operator!=** (decimal32 __lhs, decimal32 __rhs)
- bool **operator!=** (decimal32 __lhs, decimal64 __rhs)
- bool **operator!=** (decimal32 __lhs, int __rhs)
- bool **operator!=** (decimal32 __lhs, long __rhs)
- bool **operator!=** (decimal32 __lhs, long long __rhs)
- bool **operator!=** (decimal32 __lhs, unsigned int __rhs)
- bool **operator!=** (decimal32 __lhs, unsigned long __rhs)
- bool **operator!=** (decimal32 __lhs, unsigned long long __rhs)
- bool **operator!=** (decimal64 __lhs, decimal128 __rhs)
- bool **operator!=** (decimal64 __lhs, decimal32 __rhs)
- bool **operator!=** (decimal64 __lhs, decimal64 __rhs)
- bool **operator!=** (decimal64 __lhs, int __rhs)
- bool **operator!=** (decimal64 __lhs, long __rhs)
- bool **operator!=** (decimal64 __lhs, long long __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned int __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned long __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned long long __rhs)
- bool **operator!=** (int __lhs, decimal128 __rhs)
- bool **operator!=** (int __lhs, decimal32 __rhs)
- bool **operator!=** (int __lhs, decimal64 __rhs)
- bool **operator!=** (long __lhs, decimal128 __rhs)
- bool **operator!=** (long __lhs, decimal32 __rhs)
- bool **operator!=** (long __lhs, decimal64 __rhs)
- bool **operator!=** (long long __lhs, decimal128 __rhs)
- bool **operator!=** (long long __lhs, decimal32 __rhs)
- bool **operator!=** (long long __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned int __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned int __lhs, decimal32 __rhs)
- bool **operator!=** (unsigned int __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal32 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal32 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal64 __rhs)
- decimal128 **operator*** (decimal128 __lhs, decimal128 __rhs)
- decimal128 **operator*** (decimal128 __lhs, decimal32 __rhs)
- decimal128 **operator*** (decimal128 __lhs, decimal64 __rhs)
- decimal128 **operator*** (decimal128 __lhs, int __rhs)
- decimal128 **operator*** (decimal128 __lhs, long __rhs)
- decimal128 **operator*** (decimal128 __lhs, long long __rhs)
- decimal128 **operator*** (decimal128 __lhs, unsigned int __rhs)
- decimal128 **operator*** (decimal128 __lhs, unsigned long __rhs)
- decimal128 **operator*** (decimal128 __lhs, unsigned long long __rhs)
- decimal128 **operator*** (decimal32 __lhs, decimal128 __rhs)
- decimal32 **operator*** (decimal32 __lhs, decimal32 __rhs)
- decimal64 **operator*** (decimal32 __lhs, decimal64 __rhs)
- decimal32 **operator*** (decimal32 __lhs, int __rhs)
- decimal32 **operator*** (decimal32 __lhs, long __rhs)

- [decimal32 operator*](#) ([decimal32](#) __lhs, long long __rhs)
- [decimal32 operator*](#) ([decimal32](#) __lhs, unsigned int __rhs)
- [decimal32 operator*](#) ([decimal32](#) __lhs, unsigned long __rhs)
- [decimal32 operator*](#) ([decimal32](#) __lhs, unsigned long long __rhs)
- [decimal128 operator*](#) ([decimal64](#) __lhs, [decimal128](#) __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, int __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, long long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned int __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned long long __rhs)
- [decimal128 operator*](#) (int __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (int __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (int __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) (long __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (long __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (long __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) (long long __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (long long __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (long long __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) (unsigned int __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (unsigned int __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (unsigned int __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) (unsigned long __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (unsigned long __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (unsigned long __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) (unsigned long long __lhs, [decimal128](#) __rhs)
- [decimal32 operator*](#) (unsigned long long __lhs, [decimal32](#) __rhs)
- [decimal64 operator*](#) (unsigned long long __lhs, [decimal64](#) __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, [decimal128](#) __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, [decimal32](#) __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, [decimal64](#) __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, int __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, long __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, long long __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, unsigned int __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, unsigned long __rhs)
- [decimal128 operator+](#) ([decimal128](#) __lhs, unsigned long long __rhs)
- [decimal128 operator+](#) ([decimal128](#) __rhs)
- [decimal128 operator+](#) ([decimal32](#) __lhs, [decimal128](#) __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, [decimal32](#) __rhs)
- [decimal64 operator+](#) ([decimal32](#) __lhs, [decimal64](#) __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, int __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, long long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned int __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned long long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __rhs)

- **decimal128 operator+** (decimal64 __lhs, decimal128 __rhs)
- **decimal64 operator+** (decimal64 __lhs, decimal32 __rhs)
- **decimal64 operator+** (decimal64 __lhs, decimal64 __rhs)
- **decimal64 operator+** (decimal64 __lhs, int __rhs)
- **decimal64 operator+** (decimal64 __lhs, long __rhs)
- **decimal64 operator+** (decimal64 __lhs, long long __rhs)
- **decimal64 operator+** (decimal64 __lhs, unsigned int __rhs)
- **decimal64 operator+** (decimal64 __lhs, unsigned long __rhs)
- **decimal64 operator+** (decimal64 __lhs, unsigned long long __rhs)
- **decimal64 operator+** (decimal64 __rhs)
- **decimal128 operator+** (int __lhs, decimal128 __rhs)
- **decimal32 operator+** (int __lhs, decimal32 __rhs)
- **decimal64 operator+** (int __lhs, decimal64 __rhs)
- **decimal128 operator+** (long __lhs, decimal128 __rhs)
- **decimal32 operator+** (long __lhs, decimal32 __rhs)
- **decimal64 operator+** (long __lhs, decimal64 __rhs)
- **decimal128 operator+** (long long __lhs, decimal128 __rhs)
- **decimal32 operator+** (long long __lhs, decimal32 __rhs)
- **decimal64 operator+** (long long __lhs, decimal64 __rhs)
- **decimal128 operator+** (unsigned int __lhs, decimal128 __rhs)
- **decimal32 operator+** (unsigned int __lhs, decimal32 __rhs)
- **decimal64 operator+** (unsigned int __lhs, decimal64 __rhs)
- **decimal128 operator+** (unsigned long __lhs, decimal128 __rhs)
- **decimal32 operator+** (unsigned long __lhs, decimal32 __rhs)
- **decimal64 operator+** (unsigned long __lhs, decimal64 __rhs)
- **decimal128 operator+** (unsigned long long __lhs, decimal128 __rhs)
- **decimal32 operator+** (unsigned long long __lhs, decimal32 __rhs)
- **decimal64 operator+** (unsigned long long __lhs, decimal64 __rhs)
- **decimal128 operator-** (decimal128 __lhs, decimal128 __rhs)
- **decimal128 operator-** (decimal128 __lhs, decimal32 __rhs)
- **decimal128 operator-** (decimal128 __lhs, decimal64 __rhs)
- **decimal128 operator-** (decimal128 __lhs, int __rhs)
- **decimal128 operator-** (decimal128 __lhs, long __rhs)
- **decimal128 operator-** (decimal128 __lhs, long long __rhs)
- **decimal128 operator-** (decimal128 __lhs, unsigned int __rhs)
- **decimal128 operator-** (decimal128 __lhs, unsigned long __rhs)
- **decimal128 operator-** (decimal128 __lhs, unsigned long long __rhs)
- **decimal128 operator-** (decimal128 __rhs)
- **decimal128 operator-** (decimal32 __lhs, decimal128 __rhs)
- **decimal32 operator-** (decimal32 __lhs, decimal32 __rhs)
- **decimal64 operator-** (decimal32 __lhs, decimal64 __rhs)
- **decimal32 operator-** (decimal32 __lhs, int __rhs)
- **decimal32 operator-** (decimal32 __lhs, long __rhs)
- **decimal32 operator-** (decimal32 __lhs, long long __rhs)
- **decimal32 operator-** (decimal32 __lhs, unsigned int __rhs)
- **decimal32 operator-** (decimal32 __lhs, unsigned long __rhs)
- **decimal32 operator-** (decimal32 __lhs, unsigned long long __rhs)
- **decimal32 operator-** (decimal32 __rhs)
- **decimal128 operator-** (decimal64 __lhs, decimal128 __rhs)
- **decimal64 operator-** (decimal64 __lhs, decimal32 __rhs)
- **decimal64 operator-** (decimal64 __lhs, decimal64 __rhs)

- **decimal64 operator-** ([decimal64](#) __lhs, int __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, long long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned int __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned long long __rhs)
- **decimal64 operator-** ([decimal64](#) __rhs)
- **decimal128 operator-** (int __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (int __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (int __lhs, [decimal64](#) __rhs)
- **decimal128 operator-** (long __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (long __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (long __lhs, [decimal64](#) __rhs)
- **decimal128 operator-** (long long __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (long long __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (long long __lhs, [decimal64](#) __rhs)
- **decimal128 operator-** (unsigned int __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (unsigned int __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (unsigned int __lhs, [decimal64](#) __rhs)
- **decimal128 operator-** (unsigned long __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (unsigned long __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (unsigned long __lhs, [decimal64](#) __rhs)
- **decimal128 operator-** (unsigned long long __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** (unsigned long long __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** (unsigned long long __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal32](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, int __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, long __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, long long __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned int __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned long __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned long long __rhs)
- **decimal128 operator/** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** ([decimal32](#) __lhs, [decimal64](#) __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, int __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, long __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, long long __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, unsigned int __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, unsigned long __rhs)
- **decimal32 operator/** ([decimal32](#) __lhs, unsigned long long __rhs)
- **decimal128 operator/** ([decimal64](#) __lhs, [decimal128](#) __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, [decimal64](#) __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, int __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, long __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, long long __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, unsigned int __rhs)
- **decimal64 operator/** ([decimal64](#) __lhs, unsigned long __rhs)

- **decimal64 operator/** ([decimal64](#) __lhs, unsigned long long __rhs)
- **decimal128 operator/** (int __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (int __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (int __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** (long __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (long __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (long __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** (long long __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (long long __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (long long __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** (unsigned int __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (unsigned int __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (unsigned int __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** (unsigned long __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (unsigned long __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (unsigned long __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** (unsigned long long __lhs, [decimal128](#) __rhs)
- **decimal32 operator/** (unsigned long long __lhs, [decimal32](#) __rhs)
- **decimal64 operator/** (unsigned long long __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal128](#) __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal128](#) __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal128](#) __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal128](#) __lhs, int __rhs)
- **bool operator<** ([decimal128](#) __lhs, long __rhs)
- **bool operator<** ([decimal128](#) __lhs, long long __rhs)
- **bool operator<** ([decimal128](#) __lhs, unsigned int __rhs)
- **bool operator<** ([decimal128](#) __lhs, unsigned long __rhs)
- **bool operator<** ([decimal128](#) __lhs, unsigned long long __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, int __rhs)
- **bool operator<** ([decimal32](#) __lhs, long __rhs)
- **bool operator<** ([decimal32](#) __lhs, long long __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned int __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned long __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned long long __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, int __rhs)
- **bool operator<** ([decimal64](#) __lhs, long __rhs)
- **bool operator<** ([decimal64](#) __lhs, long long __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned int __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned long __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned long long __rhs)
- **bool operator<** (int __lhs, [decimal128](#) __rhs)
- **bool operator<** (int __lhs, [decimal32](#) __rhs)
- **bool operator<** (int __lhs, [decimal64](#) __rhs)
- **bool operator<** (long __lhs, [decimal128](#) __rhs)
- **bool operator<** (long __lhs, [decimal32](#) __rhs)

- bool **operator**< (long __lhs, decimal64 __rhs)
- bool **operator**< (long long __lhs, decimal128 __rhs)
- bool **operator**< (long long __lhs, decimal32 __rhs)
- bool **operator**< (long long __lhs, decimal64 __rhs)
- bool **operator**< (unsigned int __lhs, decimal128 __rhs)
- bool **operator**< (unsigned int __lhs, decimal32 __rhs)
- bool **operator**< (unsigned int __lhs, decimal64 __rhs)
- bool **operator**< (unsigned long __lhs, decimal128 __rhs)
- bool **operator**< (unsigned long __lhs, decimal32 __rhs)
- bool **operator**< (unsigned long __lhs, decimal64 __rhs)
- bool **operator**< (unsigned long long __lhs, decimal128 __rhs)
- bool **operator**< (unsigned long long __lhs, decimal32 __rhs)
- bool **operator**< (unsigned long long __lhs, decimal64 __rhs)
- bool **operator**== (decimal128 __lhs, decimal128 __rhs)
- bool **operator**== (decimal128 __lhs, decimal32 __rhs)
- bool **operator**== (decimal128 __lhs, decimal64 __rhs)
- bool **operator**== (decimal128 __lhs, int __rhs)
- bool **operator**== (decimal128 __lhs, long __rhs)
- bool **operator**== (decimal128 __lhs, long long __rhs)
- bool **operator**== (decimal128 __lhs, unsigned int __rhs)
- bool **operator**== (decimal128 __lhs, unsigned long __rhs)
- bool **operator**== (decimal128 __lhs, unsigned long long __rhs)
- bool **operator**== (decimal32 __lhs, decimal128 __rhs)
- bool **operator**== (decimal32 __lhs, decimal32 __rhs)
- bool **operator**== (decimal32 __lhs, decimal64 __rhs)
- bool **operator**== (decimal32 __lhs, int __rhs)
- bool **operator**== (decimal32 __lhs, long __rhs)
- bool **operator**== (decimal32 __lhs, long long __rhs)
- bool **operator**== (decimal32 __lhs, unsigned int __rhs)
- bool **operator**== (decimal32 __lhs, unsigned long __rhs)
- bool **operator**== (decimal32 __lhs, unsigned long long __rhs)
- bool **operator**== (decimal64 __lhs, decimal128 __rhs)
- bool **operator**== (decimal64 __lhs, decimal32 __rhs)
- bool **operator**== (decimal64 __lhs, decimal64 __rhs)
- bool **operator**== (decimal64 __lhs, int __rhs)
- bool **operator**== (decimal64 __lhs, long __rhs)
- bool **operator**== (decimal64 __lhs, long long __rhs)
- bool **operator**== (decimal64 __lhs, unsigned int __rhs)
- bool **operator**== (decimal64 __lhs, unsigned long __rhs)
- bool **operator**== (decimal64 __lhs, unsigned long long __rhs)
- bool **operator**== (int __lhs, decimal128 __rhs)
- bool **operator**== (int __lhs, decimal32 __rhs)
- bool **operator**== (int __lhs, decimal64 __rhs)
- bool **operator**== (long __lhs, decimal128 __rhs)
- bool **operator**== (long __lhs, decimal32 __rhs)
- bool **operator**== (long __lhs, decimal64 __rhs)
- bool **operator**== (long long __lhs, decimal128 __rhs)
- bool **operator**== (long long __lhs, decimal32 __rhs)
- bool **operator**== (long long __lhs, decimal64 __rhs)
- bool **operator**== (unsigned int __lhs, decimal128 __rhs)
- bool **operator**== (unsigned int __lhs, decimal32 __rhs)

- bool **operator==** (unsigned int __lhs, decimal64 __rhs)
- bool **operator==** (unsigned long __lhs, decimal128 __rhs)
- bool **operator==** (unsigned long __lhs, decimal32 __rhs)
- bool **operator==** (unsigned long __lhs, decimal64 __rhs)
- bool **operator==** (unsigned long long __lhs, decimal128 __rhs)
- bool **operator==** (unsigned long long __lhs, decimal32 __rhs)
- bool **operator==** (unsigned long long __lhs, decimal64 __rhs)
- bool **operator>** (decimal128 __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, decimal32 __rhs)
- bool **operator>** (decimal128 __lhs, decimal64 __rhs)
- bool **operator>** (decimal128 __lhs, int __rhs)
- bool **operator>** (decimal128 __lhs, long __rhs)
- bool **operator>** (decimal128 __lhs, long long __rhs)
- bool **operator>** (decimal128 __lhs, unsigned int __rhs)
- bool **operator>** (decimal128 __lhs, unsigned long __rhs)
- bool **operator>** (decimal128 __lhs, unsigned long long __rhs)
- bool **operator>** (decimal32 __lhs, decimal128 __rhs)
- bool **operator>** (decimal32 __lhs, decimal32 __rhs)
- bool **operator>** (decimal32 __lhs, decimal64 __rhs)
- bool **operator>** (decimal32 __lhs, int __rhs)
- bool **operator>** (decimal32 __lhs, long __rhs)
- bool **operator>** (decimal32 __lhs, long long __rhs)
- bool **operator>** (decimal32 __lhs, unsigned int __rhs)
- bool **operator>** (decimal32 __lhs, unsigned long __rhs)
- bool **operator>** (decimal32 __lhs, unsigned long long __rhs)
- bool **operator>** (decimal64 __lhs, decimal128 __rhs)
- bool **operator>** (decimal64 __lhs, decimal32 __rhs)
- bool **operator>** (decimal64 __lhs, decimal64 __rhs)
- bool **operator>** (decimal64 __lhs, int __rhs)
- bool **operator>** (decimal64 __lhs, long __rhs)
- bool **operator>** (decimal64 __lhs, long long __rhs)
- bool **operator>** (decimal64 __lhs, unsigned int __rhs)
- bool **operator>** (decimal64 __lhs, unsigned long __rhs)
- bool **operator>** (decimal64 __lhs, unsigned long long __rhs)
- bool **operator>** (int __lhs, decimal128 __rhs)
- bool **operator>** (int __lhs, decimal32 __rhs)
- bool **operator>** (int __lhs, decimal64 __rhs)
- bool **operator>** (long __lhs, decimal128 __rhs)
- bool **operator>** (long __lhs, decimal32 __rhs)
- bool **operator>** (long __lhs, decimal64 __rhs)
- bool **operator>** (long long __lhs, decimal128 __rhs)
- bool **operator>** (long long __lhs, decimal32 __rhs)
- bool **operator>** (long long __lhs, decimal64 __rhs)
- bool **operator>** (unsigned int __lhs, decimal128 __rhs)
- bool **operator>** (unsigned int __lhs, decimal32 __rhs)
- bool **operator>** (unsigned int __lhs, decimal64 __rhs)
- bool **operator>** (unsigned long __lhs, decimal128 __rhs)
- bool **operator>** (unsigned long __lhs, decimal32 __rhs)
- bool **operator>** (unsigned long __lhs, decimal64 __rhs)
- bool **operator>** (unsigned long long __lhs, decimal128 __rhs)
- bool **operator>** (unsigned long long __lhs, decimal32 __rhs)

- bool **operator**> (unsigned long long __lhs, decimal64 __rhs)
- bool **operator**>= (decimal128 __lhs, decimal128 __rhs)
- bool **operator**>= (decimal128 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal128 __lhs, decimal64 __rhs)
- bool **operator**>= (decimal128 __lhs, int __rhs)
- bool **operator**>= (decimal128 __lhs, long __rhs)
- bool **operator**>= (decimal128 __lhs, long long __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned long __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned long long __rhs)
- bool **operator**>= (decimal32 __lhs, decimal128 __rhs)
- bool **operator**>= (decimal32 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal32 __lhs, decimal64 __rhs)
- bool **operator**>= (decimal32 __lhs, int __rhs)
- bool **operator**>= (decimal32 __lhs, long __rhs)
- bool **operator**>= (decimal32 __lhs, long long __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned long __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned long long __rhs)
- bool **operator**>= (decimal64 __lhs, decimal128 __rhs)
- bool **operator**>= (decimal64 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal64 __lhs, decimal64 __rhs)
- bool **operator**>= (decimal64 __lhs, int __rhs)
- bool **operator**>= (decimal64 __lhs, long __rhs)
- bool **operator**>= (decimal64 __lhs, long long __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned long __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned long long __rhs)
- bool **operator**>= (int __lhs, decimal128 __rhs)
- bool **operator**>= (int __lhs, decimal32 __rhs)
- bool **operator**>= (int __lhs, decimal64 __rhs)
- bool **operator**>= (long __lhs, decimal128 __rhs)
- bool **operator**>= (long __lhs, decimal32 __rhs)
- bool **operator**>= (long __lhs, decimal64 __rhs)
- bool **operator**>= (long long __lhs, decimal128 __rhs)
- bool **operator**>= (long long __lhs, decimal32 __rhs)
- bool **operator**>= (long long __lhs, decimal64 __rhs)
- bool **operator**>= (unsigned int __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned int __lhs, decimal32 __rhs)
- bool **operator**>= (unsigned int __lhs, decimal64 __rhs)
- bool **operator**>= (unsigned long __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned long __lhs, decimal32 __rhs)
- bool **operator**>= (unsigned long __lhs, decimal64 __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal32 __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal64 __rhs)

4.15.1 Detailed Description

ISO/IEC TR 24733 Decimal floating-point arithmetic.

4.15.2 Function Documentation

decimal32_to_long_long()

```
long long std::decimal::decimal32_to_long_long (
    decimal32 __d )
```

Non-conforming extension: Conversion to integral type.

4.16 std::experimental Namespace Reference

Classes

- class [any](#)
- class [bad_any_cast](#)
- class [bad_optional_access](#)
- class [basic_string_view](#)
- struct [in_place_t](#)
- struct [nullopt_t](#)
- class [optional](#)
- class [ostream_joiner](#)
- struct [owner_less< shared_ptr< _Tp > >](#)
- struct [owner_less< weak_ptr< _Tp > >](#)
- class [propagate_const](#)

Typedefs

- template<typename _RAIter, typename _Hash, typename _Pred, typename _Val = typename iterator_traits<_RAIter>::value_type, typename _Diff = typename iterator_traits<_RAIter>::difference_type>
using [__boyer_moore_base_t](#) = std::__conditional_t< std::__is_byte_like< _Val, _Pred >::value, __boyer_moore_array_base< _Diff, 256, _Pred >, __boyer_moore_map_base< _Val, _Diff, _Hash, _Pred > >
- using [erased_type](#) = std::__erased_type
- using [string_view](#) = [basic_string_view](#)< char >
- using [u16string_view](#) = [basic_string_view](#)< char16_t >
- using [u32string_view](#) = [basic_string_view](#)< char32_t >
- using [wstring_view](#) = [basic_string_view](#)< wchar_t >

Functions

- template<typename _Fn, typename _Tuple, std::size_t... _Idx>
constexpr decltype(auto) [__apply_impl](#) (_Fn &&__f, _Tuple &&__t, [std::index_sequence](#)< _Idx... >)
- template<typename _Tp, size_t _Nm, size_t... _Idx>
constexpr [array](#)< [remove_cv_t](#)< _Tp >, _Nm > [__to_array](#) (_Tp(&__a)[_Nm], [index_sequence](#)< _Idx... >)
- [std::default_random_engine](#) & [_S_randint_engine](#) ()
- template<typename _ValueType >
_ValueType [any_cast](#) (const [any](#) &__any)
- template<typename _Fn, typename _Tuple >
constexpr decltype(auto) [apply](#) (_Fn &&__f, _Tuple &&__t)
- template<typename _Tp >
bool [atomic_compare_exchange_strong](#) (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)
- template<typename _Tp >
bool [atomic_compare_exchange_strong_explicit](#) (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, [memory_order](#) __success, [memory_order](#) __failure)

- `template<typename _Tp >`
`bool atomic_compare_exchange_weak (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp >`
`> __w)`
- `template<typename _Tp >`
`bool atomic_compare_exchange_weak_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v,`
`shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`void atomic_exchange (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp >`
`shared_ptr< _Tp > atomic_exchange_explicit (const shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r,`
`memory_order __mo)`
- `template<typename _Tp >`
`bool atomic_is_lock_free (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > atomic_load (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > atomic_load_explicit (const shared_ptr< _Tp > *__p, memory_order __mo)`
- `template<typename _Tp >`
`void atomic_store (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp >`
`shared_ptr< _Tp > atomic_store_explicit (const shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r,`
`memory_order __mo)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > const_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > dynamic_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Up >`
`void erase (basic_string< _CharT, _Traits, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`void erase (deque< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`void erase (forward_list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`void erase (list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Up >`
`void erase (vector< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Predicate >`
`void erase_if (basic_string< _CharT, _Traits, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void erase_if (deque< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void erase_if (forward_list< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void erase_if (list< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`void erase_if (map< _Key, _Tp, _Compare, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`void erase_if (multimap< _Key, _Tp, _Compare, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`void erase_if (multiset< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`void erase_if (set< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`

- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`void erase_if (unordered_map< _Key, _Tp, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`void erase_if (unordered_multimap< _Key, _Tp, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`void erase_if (unordered_multiset< _Key, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`void erase_if (unordered_set< _Key, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void erase_if (vector< _Tp, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > gcd (_Mn __m, _Nn __n) noexcept`
- `template<typename _Del, typename _Tp >`
`_Del * get_deleter (const shared_ptr< _Tp > &__p) noexcept`
- `template<typename _Tp >`
`constexpr const _Tp & get_underlying (const propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Tp >`
`constexpr _Tp & get_underlying (propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > lcm (_Mn __m, _Nn __n)`
- `template<typename _Dest = void, typename... _Types>`
`constexpr array< typename __make_array_elem< _Dest, _Types... >::type, sizeof...(_Types)> make_array (←
_Types &&... __t)`
- `template<typename _RAlter, typename _Hash = std::hash<typename std::iterator_traits<_RAlter>::value_type>, typename _Binary←
Predicate = equal_to<>>`
`boyer_moore_horspool_searcher< _RAlter, _Hash, _BinaryPredicate > make_boyer_moore_horspool_searcher`
`(_RAlter __pat_first, _RAlter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _RAlter, typename _Hash = std::hash<typename std::iterator_traits<_RAlter>::value_type>, typename _Binary←
Predicate = equal_to<>>`
`boyer_moore_searcher< _RAlter, _Hash, _BinaryPredicate > make_boyer_moore_searcher (_RAlter __pat ←
first, _RAlter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _ForwardIterator, typename _BinaryPredicate = std::equal_to<>>`
`default_searcher< _ForwardIterator, _BinaryPredicate > make_default_searcher (_ForwardIterator __pat_first,`
`_ForwardIterator __pat_last, _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _Tp >`
`observer_ptr< _Tp > make_observer (_Tp *__p) noexcept`
- `template<typename _CharT, typename _Traits, typename _DelimT >`
`ostream_joiner< decay_t< _DelimT >, _CharT, _Traits > make_ostream_joiner (basic_ostream< _CharT, _←
Traits > &__os, _DelimT &&__delimiter)`
- `template<typename _Fn >`
`auto not_fn (_Fn &&__fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn && >::value)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<`
`_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator!= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator!= (const _Tp &__t, const propagate_const< _Up > &__pu)`

- `template<typename _Tp, typename _Up >`
`constexpr bool operator!= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator!= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool operator!= (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`bool operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`constexpr bool operator!= (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp >`
`bool operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp >`
`bool operator!= (nullptr_t, observer_ptr< _Tp > __p) noexcept`
- `template<typename _Tp >`
`bool operator!= (observer_ptr< _Tp > __p, nullptr_t) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator!= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `constexpr basic_string_view< char > operator""sv (const char *__str, size_t __len) noexcept`
- `constexpr basic_string_view< char16_t > operator""sv (const char16_t *__str, size_t __len) noexcept`
- `constexpr basic_string_view< char32_t > operator""sv (const char32_t *__str, size_t __len) noexcept`
- `constexpr basic_string_view< wchar_t > operator""sv (const wchar_t *__str, size_t __len) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator< (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator< (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator< (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator< (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`bool operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator< (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, basic_string_view< _CharT, _Traits > __str)`
- `template<typename _Ch, typename _Tr, typename _Tp >`
`std::basic_ostream< _Ch, _Tr > & operator<< (std::basic_ostream< _Ch, _Tr > &__os, const shared_ptr< _Tp > &__p)`

- `template<typename _CharT, typename _Traits >`
`constexpr bool operator<= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator<= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<`
`_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator<= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator<= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator<= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator<= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`bool operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator<= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view<`
`_CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator== (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits >`
`__y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator== (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator== (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator== (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool operator== (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`bool operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`constexpr bool operator== (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp >`
`bool operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp >`
`bool operator== (nullptr_t, observer_ptr< _Tp > __p) noexcept`
- `template<typename _Tp >`
`bool operator== (observer_ptr< _Tp > __p, nullptr_t) noexcept`

- `template<typename _Tp, typename _Up >`
`bool operator== (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator> (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator> (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator> (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator> (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`bool operator> (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator> (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator> (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool operator>= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator>= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator>= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator>= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`bool operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator>= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _IntType >`
`_IntType randint (_IntType __a, _IntType __b)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > reinterpret_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `void reseed ()`

- void **reseed** (default_random_engine::result_type __value)
- template<typename _PopulationIterator, typename _SampleIterator, typename _Distance >
_SampleIterator **sample** (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n)
- template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomNumberGenerator >
_SampleIterator **sample** (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomNumberGenerator &&__g)
- template<typename _ForwardIterator, typename _Searcher >
_ForwardIterator **search** (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &__searcher)
- template<typename _RandomAccessIterator >
void **shuffle** (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _Tp, typename _Tp1 >
shared_ptr< _Tp > **static_pointer_cast** (const shared_ptr< _Tp1 > &__r) noexcept
- void **swap** (any &__x, any &__y) noexcept
- template<typename _Tp >
void **swap** (observer_ptr< _Tp > &__p1, observer_ptr< _Tp > &__p2) noexcept
- template<typename _Tp >
constexpr enable_if_t< __is_swappable< _Tp >::value, void > **swap** (propagate_const< _Tp > &__pt, propagate_const< _Tp > &__pt2) noexcept(__is_nothrow_swappable< _Tp >::value)
- template<typename _Tp >
void **swap** (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept
- template<typename _Tp >
void **swap** (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept
- template<typename _Tp, size_t _Nm>
constexpr array< remove_cv_t< _Tp >, _Nm > **to_array** (_Tp(&__a)[_Nm]) noexcept(is_nothrow_constructible< remove_cv_t< _Tp >, _Tp &::value)
- template<typename _ValueType >
_ValueType **any_cast** (any &__any)
- template<typename _ValueType, typename enable_if< !is_move_constructible< _ValueType >::value || is_lvalue_reference< _ValueType >::value, bool >::type = true>
_ValueType **any_cast** (any &&__any)
- template<typename _ValueType, typename enable_if< is_move_constructible< _ValueType >::value && !is_lvalue_reference< _ValueType >::value, bool >::type = false>
_ValueType **any_cast** (any &&__any)
- template<typename _ValueType >
const _ValueType * **any_cast** (const any * __any) noexcept
- template<typename _ValueType >
_ValueType * **any_cast** (any * __any) noexcept

Variables

- template<typename _Yp, typename _Tp >
constexpr bool **__sp_compatible_v**
- template<typename _Tp, typename _Yp >
constexpr bool **__sp_is_constructible_v**
- constexpr **in_place_t** in_place
- template<typename _Tp >
constexpr bool **is_bind_expression_v**

- `template<typename _Tp >`
`constexpr bool is_error_code_enum_v`
- `template<typename _Tp >`
`constexpr bool is_error_condition_enum_v`
- `template<typename _Tp >`
`constexpr int is_placeholder_v`
- `constexpr nullopt_t nullopt`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_greater_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_greater_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_less_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_less_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool ratio_not_equal_v`
- `template<typename _Tp >`
`constexpr size_t tuple_size_v`
- `template<typename _Tp >`
`constexpr bool is_void_v`
- `template<typename _Tp >`
`constexpr bool is_null_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_integral_v`
- `template<typename _Tp >`
`constexpr bool is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool is_array_v`
- `template<typename _Tp >`
`constexpr bool is_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool is_rvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_enum_v`
- `template<typename _Tp >`
`constexpr bool is_union_v`
- `template<typename _Tp >`
`constexpr bool is_class_v`
- `template<typename _Tp >`
`constexpr bool is_function_v`
- `template<typename _Tp >`
`constexpr bool is_reference_v`

- `template<typename _Tp >`
`constexpr bool is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool is_object_v`
- `template<typename _Tp >`
`constexpr bool is_scalar_v`
- `template<typename _Tp >`
`constexpr bool is_compound_v`
- `template<typename _Tp >`
`constexpr bool is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool is_const_v`
- `template<typename _Tp >`
`constexpr bool is_volatile_v`
- `template<typename _Tp >`
`constexpr bool is_trivial_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool is_pod_v`
- `template<typename _Tp >`
`constexpr bool is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool is_empty_v`
- `template<typename _Tp >`
`constexpr bool is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool is_abstract_v`
- `template<typename _Tp >`
`constexpr bool is_final_v`
- `template<typename _Tp >`
`constexpr bool is_signed_v`
- `template<typename _Tp >`
`constexpr bool is_unsigned_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_move_assignable_v`

- `template<typename _Tp >`
`constexpr bool is_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_trivially_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_trivially_destructible_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_nothrow_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr size_t alignment_of_v`
- `template<typename _Tp >`
`constexpr size_t rank_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t extent_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool is_same_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool is_base_of_v`
- `template<typename _From, typename _To >`
`constexpr bool is_convertible_v`

- `template<typename... _Bn>`
`constexpr bool conjunction_v`

- `template<typename... _Bn>`
`constexpr bool disjunction_v`
- `template<typename _Pp >`
`constexpr bool negation_v`
- `template<typename... >`
`using void_t = void`
- `template<template< typename... > class _Op, typename... _Args>`
`using is_detected = typename std::__detector< nonesuch, void, _Op, _Args... >::value_t`
- `template<template< typename... > class _Op, typename... _Args>`
`using detected_t = typename std::__detector< nonesuch, void, _Op, _Args... >::type`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
`using detected_or = std::__detected_or< _Default, _Op, _Args... >`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
`using detected_or_t = typename detected_or< _Default, _Op, _Args... >::type`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
`using is_detected_exact = is_same< _Expected, detected_t< _Op, _Args... > >`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
`using is_detected_convertible = is_convertible< detected_t< _Op, _Args... >, _To >`
- `template<template< typename... > class _Op, typename... _Args>`
`constexpr bool is_detected_v`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
`constexpr bool is_detected_exact_v`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
`constexpr bool is_detected_convertible_v`

4.16.1 Detailed Description

Namespace for features defined in ISO Technical Specifications.

4.16.2 Function Documentation

gcd()

```
template<typename _Mn, typename _Nn >
constexpr common_type_t< _Mn, _Nn > std::experimental::fundamentals_v2::gcd (
    _Mn __m,
    _Nn __n ) [constexpr], [noexcept]
```

Greatest common divisor.

get_deleter()

```
template<typename _Del, typename _Tp >
_Del * std::experimental::fundamentals_v2::get_deleter (
    const shared_ptr< _Tp > & __p ) [inline], [noexcept]
```

C++14 20.8.2.2.10.

lcm()

```
template<typename _Mn, typename _Nn >
constexpr common_type_t< _Mn, _Nn > std::experimental::fundamentals_v2::lcm (
    _Mn __m,
    _Nn __n ) [constexpr]
```

Least common multiple.

make_boyer_moore_horspool_searcher()

```
template<typename _RAIter , typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>
boyer_moore_horspool_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::fundamentals_v1::make_boyer_moore_horspool_searcher (
    _RAIter __pat_first,
    _RAIter __pat_last,
    _Hash __hf = _Hash(),
    _BinaryPredicate __pred = _BinaryPredicate() ) [inline]
```

Generator function for boyer_moore_horspool_searcher.

make_boyer_moore_searcher()

```
template<typename _RAIter , typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>
boyer_moore_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::fundamentals_v1::make_boyer_moore_searcher (
    _RAIter __pat_first,
    _RAIter __pat_last,
    _Hash __hf = _Hash(),
    _BinaryPredicate __pred = _BinaryPredicate() ) [inline]
```

Generator function for boyer_moore_searcher.

make_default_searcher()

```
template<typename _ForwardIterator , typename _BinaryPredicate = std::equal_to<>>
default_searcher< _ForwardIterator, _BinaryPredicate > std::experimental::fundamentals_v1::make_default_searcher (
    _ForwardIterator __pat_first,
    _ForwardIterator __pat_last,
    _BinaryPredicate __pred = _BinaryPredicate() ) [inline]
```

Generator function for default_searcher.

make_ostream_joiner()

```
template<typename _CharT , typename _Traits , typename _DelimT >
ostream_joiner< decay_t< _DelimT >, _CharT, _Traits > std::experimental::fundamentals_v2::make_ostream_joiner (
    basic_ostream< _CharT, _Traits > & __os,
    _DelimT && __delimiter ) [inline]
```

Object generator for ostream_joiner.

not_fn()

```
template<typename _Fn >
auto std::experimental::fundamentals_v2::not_fn (
    _Fn && __fn ) [inline], [noexcept]
```

[func.not_fn] Function template not_fn

sample()

```
template<typename _PopulationIterator , typename _SampleIterator , typename _Distance , typename _UniformRandomNumberGenerator >
_SampleIterator std::experimental::fundamentals_v2::sample (
```

```
    _PopulationIterator __first,  
    _PopulationIterator __last,  
    _SampleIterator __out,  
    _Distance __n,  
    _UniformRandomNumberGenerator && __g )
```

Take a random sample from a population.

4.16.3 Variable Documentation

`is_bind_expression_v`

```
template<typename _Tp >  
constexpr bool std::experimental::fundamentals_v1::is_bind_expression_v [constexpr]  
Variable template for std::is_bind_expression.
```

`is_placeholder_v`

```
template<typename _Tp >  
constexpr int std::experimental::fundamentals_v1::is_placeholder_v [constexpr]  
Variable template for std::is_placeholder.
```

4.17 `std::filesystem` Namespace Reference

Classes

- struct [__directory_iterator_proxy](#)
- class [directory_entry](#)
- class [directory_iterator](#)
- class [file_status](#)
- class [filesystem_error](#)
- class [path](#)
- class [recursive_directory_iterator](#)
- struct [space_info](#)

Typedefs

- using [file_time_type](#) = [__file_clock::time_point](#)

Enumerations

- enum class [copy_options](#) : unsigned short { **none** , **skip_existing** , **overwrite_existing** , **update_existing** , **recursive** , **copy_symlinks** , **skip_symlinks** , **directories_only** , **create_symlinks** , **create_hard_links** }
- enum class [directory_options](#) : unsigned char { **none** , **follow_directory_symlink** , **skip_permission_denied** }
- enum class [file_type](#) : signed char { **none** , **not_found** , **regular** , **directory** , **symlink** , **block** , **character** , **fifo** , **socket** , **unknown** }
- enum class [perm_options](#) : unsigned { **replace** , **add** , **remove** , **nofollow** }
- enum class [perms](#) : unsigned { **none** , **owner_read** , **owner_write** , **owner_exec** , **owner_all** , **group_read** , **group_write** , **group_exec** , **group_all** , **others_read** , **others_write** , **others_exec** , **others_all** , **all** , **set_uid** , **set_gid** , **sticky_bit** , **mask** , **unknown** }

Functions

- **path absolute** (const [path](#) &__p)
- **path absolute** (const [path](#) &__p, [error_code](#) &__ec)
- **path canonical** (const [path](#) &__p)
- **path canonical** (const [path](#) &__p, [error_code](#) &__ec)
- void **copy** (const [path](#) &__from, const [path](#) &__to)
- void **copy** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options)
- void **copy** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options, [error_code](#) &__ec)
- void **copy** (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec)
- bool **copy_file** (const [path](#) &__from, const [path](#) &__to)
- bool **copy_file** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option)
- bool **copy_file** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option, [error_code](#) &__ec)
- bool **copy_file** (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec)
- void **copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_symlink)
- void **copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- bool **create_directories** (const [path](#) &__p)
- bool **create_directories** (const [path](#) &__p, [error_code](#) &__ec)
- bool **create_directory** (const [path](#) &__p)
- bool **create_directory** (const [path](#) &__p, const [path](#) &__attributes)
- bool **create_directory** (const [path](#) &__p, const [path](#) &__attributes, [error_code](#) &__ec) noexcept
- bool **create_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- void **create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- void **create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link)
- void **create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link, [error_code](#) &__ec) noexcept
- void **create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- **path current_path** ()
- void **current_path** (const [path](#) &__p)
- void **current_path** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- **path current_path** ([error_code](#) &__ec)
- bool **equivalent** (const [path](#) &__p1, const [path](#) &__p2)
- bool **equivalent** (const [path](#) &__p1, const [path](#) &__p2, [error_code](#) &__ec) noexcept
- bool **exists** (const [path](#) &__p)
- bool **exists** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **exists** ([file_status](#)) noexcept
- uintmax_t **file_size** (const [path](#) &__p)
- uintmax_t **file_size** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- uintmax_t **hard_link_count** (const [path](#) &__p)
- uintmax_t **hard_link_count** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- size_t **hash_value** (const [path](#) &__p) noexcept
- bool **is_block_file** (const [path](#) &__p)
- bool **is_block_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **is_block_file** ([file_status](#) __s) noexcept
- bool **is_character_file** (const [path](#) &__p)
- bool **is_character_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **is_character_file** ([file_status](#) __s) noexcept
- bool **is_directory** (const [path](#) &__p)
- bool **is_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **is_directory** ([file_status](#) __s) noexcept

- `bool is_empty (const path &__p)`
- `bool is_empty (const path &__p, error_code &__ec)`
- `bool is_fifo (const path &__p)`
- `bool is_fifo (const path &__p, error_code &__ec) noexcept`
- `bool is_fifo (file_status __s) noexcept`
- `bool is_other (const path &__p)`
- `bool is_other (const path &__p, error_code &__ec) noexcept`
- `bool is_other (file_status) noexcept`
- `bool is_regular_file (const path &__p)`
- `bool is_regular_file (const path &__p, error_code &__ec) noexcept`
- `bool is_regular_file (file_status) noexcept`
- `bool is_socket (const path &__p)`
- `bool is_socket (const path &__p, error_code &__ec) noexcept`
- `bool is_socket (file_status __s) noexcept`
- `bool is_symlink (const path &__p)`
- `bool is_symlink (const path &__p, error_code &__ec) noexcept`
- `bool is_symlink (file_status) noexcept`
- `file_time_type last_write_time (const path &)`
- `file_time_type last_write_time (const path &, error_code &) noexcept`
- `void last_write_time (const path &__p, file_time_type __new_time)`
- `void last_write_time (const path &__p, file_time_type __new_time, error_code &__ec) noexcept`
- `copy_options & operator&= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options operator^ (copy_options __x, copy_options __y) noexcept`
- `copy_options & operator^= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options operator| (copy_options __x, copy_options __y) noexcept`
- `copy_options & operator|= (copy_options &__x, copy_options __y) noexcept`
- `constexpr copy_options operator~ (copy_options __x) noexcept`
- `void permissions (const path &, perms, perm_options, error_code &) noexcept`
- `void permissions (const path &__p, perms __prms, error_code &__ec) noexcept`
- `void permissions (const path &__p, perms __prms, perm_options __opts=perm_options::replace)`
- `path proximate (const path &__p, const path &__base, error_code &__ec)`
- `path proximate (const path &__p, const path &__base=current_path())`
- `path proximate (const path &__p, error_code &__ec)`
- `path read_symlink (const path &__p)`
- `path read_symlink (const path &__p, error_code &__ec)`
- `path relative (const path &__p, const path &__base, error_code &__ec)`
- `path relative (const path &__p, const path &__base=current_path())`
- `path relative (const path &__p, error_code &__ec)`
- `bool remove (const path &, error_code &) noexcept`
- `bool remove (const path &__p)`
- `uintmax_t remove_all (const path &)`
- `uintmax_t remove_all (const path &, error_code &)`
- `void rename (const path &__from, const path &__to)`
- `void rename (const path &__from, const path &__to, error_code &__ec) noexcept`
- `void resize_file (const path &__p, uintmax_t __size)`
- `void resize_file (const path &__p, uintmax_t __size, error_code &__ec) noexcept`
- `space_info space (const path &__p)`
- `space_info space (const path &__p, error_code &__ec) noexcept`
- `file_status status (const path &)`
- `file_status status (const path &, error_code &) noexcept`
- `bool status_known (file_status) noexcept`

- `file_status symlink_status` (const `path` &)
 - `file_status symlink_status` (const `path` &, `error_code` &) noexcept
 - `path temp_directory_path` ()
 - `path temp_directory_path` (`error_code` & __ec)
 - `path weakly_canonical` (const `path` & __p)
 - `path weakly_canonical` (const `path` & __p, `error_code` & __ec)
-
- constexpr `perms operator|` (`perms` __x, `perms` __y) noexcept
 - constexpr `perms operator^` (`perms` __x, `perms` __y) noexcept
 - constexpr `perms operator~` (`perms` __x) noexcept
 - `perms & operator&=` (`perms` & __x, `perms` __y) noexcept
 - `perms & operator|=` (`perms` & __x, `perms` __y) noexcept
 - `perms & operator^=` (`perms` & __x, `perms` __y) noexcept
-
- constexpr `perm_options operator|` (`perm_options` __x, `perm_options` __y) noexcept
 - constexpr `perm_options operator^` (`perm_options` __x, `perm_options` __y) noexcept
 - constexpr `perm_options operator~` (`perm_options` __x) noexcept
 - `perm_options & operator&=` (`perm_options` & __x, `perm_options` __y) noexcept
 - `perm_options & operator|=` (`perm_options` & __x, `perm_options` __y) noexcept
 - `perm_options & operator^=` (`perm_options` & __x, `perm_options` __y) noexcept
-
- constexpr `directory_options operator|` (`directory_options` __x, `directory_options` __y) noexcept
 - constexpr `directory_options operator^` (`directory_options` __x, `directory_options` __y) noexcept
 - constexpr `directory_options operator~` (`directory_options` __x) noexcept
 - `directory_options & operator&=` (`directory_options` & __x, `directory_options` __y) noexcept
 - `directory_options & operator|=` (`directory_options` & __x, `directory_options` __y) noexcept
 - `directory_options & operator^=` (`directory_options` & __x, `directory_options` __y) noexcept

4.17.1 Detailed Description

ISO C++ 2017 namespace for File System library.

4.18 std::literals Namespace Reference

Namespaces

- namespace `chrono_literals`

Functions

- constexpr `chrono::day operator""d` (unsigned long long __d) noexcept
- template<char... _Digits>
constexpr `chrono::hours operator""h` ()
- constexpr `chrono::duration`< long double, `ratio`< 3600, 1 > > `operator""h` (long double __hours)
- constexpr `std::complex`< double > `operator""i` (long double __num)
- constexpr `std::complex`< double > `operator""i` (unsigned long long __num)
- constexpr `std::complex`< float > `operator""if` (long double __num)
- constexpr `std::complex`< float > `operator""if` (unsigned long long __num)
- constexpr `std::complex`< long double > `operator""il` (long double __num)

- constexpr `std::complex`< long double > **operator**""il (unsigned long long __num)
- template<char... _Digits>
constexpr `chrono::minutes` **operator**""min ()
- constexpr `chrono::duration`< long double, `ratio`< 60, 1 > > **operator**""min (long double __mins)
- template<char... _Digits>
constexpr `chrono::milliseconds` **operator**""ms ()
- constexpr `chrono::duration`< long double, `milli` > **operator**""ms (long double __msecs)
- template<char... _Digits>
constexpr `chrono::nanoseconds` **operator**""ns ()
- constexpr `chrono::duration`< long double, `nano` > **operator**""ns (long double __nsecs)
- template<char... _Digits>
constexpr `chrono::seconds` **operator**""s ()
- `basic_string`< char > **operator**""s (const char * __str, size_t __len)
- `basic_string`< char16_t > **operator**""s (const char16_t * __str, size_t __len)
- `basic_string`< char32_t > **operator**""s (const char32_t * __str, size_t __len)
- `basic_string`< wchar_t > **operator**""s (const wchar_t * __str, size_t __len)
- constexpr `chrono::duration`< long double > **operator**""s (long double __secs)
- constexpr `basic_string_view`< char > **operator**""sv (const char * __str, size_t __len) noexcept
- constexpr `basic_string_view`< char16_t > **operator**""sv (const char16_t * __str, size_t __len) noexcept
- constexpr `basic_string_view`< char32_t > **operator**""sv (const char32_t * __str, size_t __len) noexcept
- constexpr `basic_string_view`< wchar_t > **operator**""sv (const wchar_t * __str, size_t __len) noexcept
- template<char... _Digits>
constexpr `chrono::microseconds` **operator**""us ()
- constexpr `chrono::duration`< long double, `micro` > **operator**""us (long double __usecs)
- constexpr `chrono::year` **operator**""y (unsigned long long __y) noexcept

4.18.1 Detailed Description

ISO C++ inline namespace for literal suffixes.

4.19 `std::literals::chrono_literals` Namespace Reference

Functions

- constexpr `chrono::day` **operator**""d (unsigned long long __d) noexcept
- template<char... _Digits>
constexpr `chrono::hours` **operator**""h ()
- constexpr `chrono::duration`< long double, `ratio`< 3600, 1 > > **operator**""h (long double __hours)
- template<char... _Digits>
constexpr `chrono::minutes` **operator**""min ()
- constexpr `chrono::duration`< long double, `ratio`< 60, 1 > > **operator**""min (long double __mins)
- template<char... _Digits>
constexpr `chrono::milliseconds` **operator**""ms ()
- constexpr `chrono::duration`< long double, `milli` > **operator**""ms (long double __msecs)
- template<char... _Digits>
constexpr `chrono::nanoseconds` **operator**""ns ()
- constexpr `chrono::duration`< long double, `nano` > **operator**""ns (long double __nsecs)
- template<char... _Digits>
constexpr `chrono::seconds` **operator**""s ()
- constexpr `chrono::duration`< long double > **operator**""s (long double __secs)
- template<char... _Digits>
constexpr `chrono::microseconds` **operator**""us ()
- constexpr `chrono::duration`< long double, `micro` > **operator**""us (long double __usecs)
- constexpr `chrono::year` **operator**""y (unsigned long long __y) noexcept

4.19.1 Detailed Description

ISO C++ 2014 namespace for suffixes for duration literals.

These suffixes can be used to create `chrono::duration` values with tick periods of hours, minutes, seconds, milliseconds, microseconds or nanoseconds. For example, `std::chrono::seconds(5)` can be written as `5s` after making the suffix visible in the current scope. The suffixes can be made visible by a `using-directive` or `using-declaration` such as:

- `using namespace std::chrono_literals;`
- `using namespace std::literals;`
- `using namespace std::chrono;`
- `using namespace std;`
- `using std::chrono_literals::operator""s;`

The result of these suffixes on an integer literal is one of the standard typedefs such as `std::chrono::hours`. The result on a floating-point literal is a duration type with the specified tick period and an unspecified floating-point representation, for example `1.5e2ms` might be equivalent to `chrono::duration<long double, chrono::milli>(1.5e2)`.

Since

C+14

4.20 `std::numbers` Namespace Reference

Variables

- constexpr double **e**
- template<typename _Tp >
constexpr _Tp **e_v**
- constexpr double **egamma**
- template<typename _Tp >
constexpr _Tp **egamma_v**
- constexpr double **inv_pi**
- template<typename _Tp >
constexpr _Tp **inv_pi_v**
- constexpr double **inv_sqrt3**
- template<typename _Tp >
constexpr _Tp **inv_sqrt3_v**
- constexpr double **inv_sqrtpi**
- template<typename _Tp >
constexpr _Tp **inv_sqrtpi_v**
- constexpr double **ln10**
- template<typename _Tp >
constexpr _Tp **ln10_v**
- constexpr double **ln2**
- template<typename _Tp >
constexpr _Tp **ln2_v**
- constexpr double **log10e**
- template<typename _Tp >
constexpr _Tp **log10e_v**
- constexpr double **log2e**

- `template<typename _Tp >`
`constexpr _Tp log2e_v`
- `constexpr double phi`
- `template<typename _Tp >`
`constexpr _Tp phi_v`
- `constexpr double pi`
- `template<typename _Tp >`
`constexpr _Tp pi_v`
- `constexpr double sqrt2`
- `template<typename _Tp >`
`constexpr _Tp sqrt2_v`
- `constexpr double sqrt3`
- `template<typename _Tp >`
`constexpr _Tp sqrt3_v`

4.20.1 Detailed Description

Namespace for mathematical constants.

4.20.2 Variable Documentation

e_v

```
template<typename _Tp >
constexpr _Tp std::numbers::e_v [inline], [constexpr]
e
```

egamma_v

```
template<typename _Tp >
constexpr _Tp std::numbers::egamma_v [inline], [constexpr]
The Euler-Mascheroni constant.
```

inv_pi_v

```
template<typename _Tp >
constexpr _Tp std::numbers::inv_pi_v [inline], [constexpr]
1/pi
```

inv_sqrt3_v

```
template<typename _Tp >
constexpr _Tp std::numbers::inv_sqrt3_v [inline], [constexpr]
1/sqrt(3)
```

inv_sqrtpi_v

```
template<typename _Tp >
constexpr _Tp std::numbers::inv_sqrtpi_v [inline], [constexpr]
1/sqrt(pi)
```

ln10_v

```
template<typename _Tp >
constexpr _Tp std::numbers::ln10_v [inline], [constexpr]
log_e 10
```

ln2_v

```
template<typename _Tp >
constexpr _Tp std::numbers::ln2_v [inline], [constexpr]
log_e 2
```

log10e_v

```
template<typename _Tp >
constexpr _Tp std::numbers::log10e_v [inline], [constexpr]
log_10 e
```

log2e_v

```
template<typename _Tp >
constexpr _Tp std::numbers::log2e_v [inline], [constexpr]
log_2 e
```

phi_v

```
template<typename _Tp >
constexpr _Tp std::numbers::phi_v [inline], [constexpr]
The golden ratio, (1+sqrt(5))/2.
```

pi_v

```
template<typename _Tp >
constexpr _Tp std::numbers::pi_v [inline], [constexpr]
pi
```

sqrt2_v

```
template<typename _Tp >
constexpr _Tp std::numbers::sqrt2_v [inline], [constexpr]
sqrt(2)
```

sqrt3_v

```
template<typename _Tp >
constexpr _Tp std::numbers::sqrt3_v [inline], [constexpr]
sqrt(3)
```

4.21 std::placeholders Namespace Reference**Variables**

- const [_Placeholder](#)< 1 > [_1](#)
- const [_Placeholder](#)< 10 > [_10](#)
- const [_Placeholder](#)< 11 > [_11](#)
- const [_Placeholder](#)< 12 > [_12](#)
- const [_Placeholder](#)< 13 > [_13](#)
- const [_Placeholder](#)< 14 > [_14](#)
- const [_Placeholder](#)< 15 > [_15](#)
- const [_Placeholder](#)< 16 > [_16](#)
- const [_Placeholder](#)< 17 > [_17](#)

- `const _Placeholder< 18 > _18`
- `const _Placeholder< 19 > _19`
- `const _Placeholder< 2 > _2`
- `const _Placeholder< 20 > _20`
- `const _Placeholder< 21 > _21`
- `const _Placeholder< 22 > _22`
- `const _Placeholder< 23 > _23`
- `const _Placeholder< 24 > _24`
- `const _Placeholder< 25 > _25`
- `const _Placeholder< 26 > _26`
- `const _Placeholder< 27 > _27`
- `const _Placeholder< 28 > _28`
- `const _Placeholder< 29 > _29`
- `const _Placeholder< 3 > _3`
- `const _Placeholder< 4 > _4`
- `const _Placeholder< 5 > _5`
- `const _Placeholder< 6 > _6`
- `const _Placeholder< 7 > _7`
- `const _Placeholder< 8 > _8`
- `const _Placeholder< 9 > _9`

4.21.1 Detailed Description

ISO C++ 2011 namespace for `std::bind` placeholders.

Since

C++11

4.22 `std::regex_constants` Namespace Reference

5.1 Regular Expression Syntax Options

- enum `syntax_option_type` : unsigned int {
 `_S_icase`, `_S_nosubs`, `_S_optimize`, `_S_collate`,
 `_S_ECMAScript`, `_S_basic`, `_S_extended`, `_S_awk`,
 `_S_grep`, `_S_egrep`, `_S_polynomial`, `_S_multiline` }
- constexpr `syntax_option_type icase`
- constexpr `syntax_option_type nosubs`
- constexpr `syntax_option_type optimize`
- constexpr `syntax_option_type collate`
- constexpr `syntax_option_type ECMAScript`
- constexpr `syntax_option_type basic`
- constexpr `syntax_option_type extended`
- constexpr `syntax_option_type awk`
- constexpr `syntax_option_type grep`
- constexpr `syntax_option_type egrep`
- constexpr `syntax_option_type multiline`
- constexpr `syntax_option_type __multiline`
- constexpr `syntax_option_type __polynomial`
- constexpr `syntax_option_type operator&` (`syntax_option_type __a`, `syntax_option_type __b`)
- constexpr `syntax_option_type operator|` (`syntax_option_type __a`, `syntax_option_type __b`)
- constexpr `syntax_option_type operator^` (`syntax_option_type __a`, `syntax_option_type __b`)

- constexpr [syntax_option_type](#) operator~ (syntax_option_type __a)
- constexpr [syntax_option_type](#) & operator&= (syntax_option_type &__a, syntax_option_type __b)
- constexpr [syntax_option_type](#) & operator|= (syntax_option_type &__a, syntax_option_type __b)
- constexpr [syntax_option_type](#) & operator^= (syntax_option_type &__a, syntax_option_type __b)

5.2 Matching Rules

Matching a regular expression against a sequence of characters [first, last) proceeds according to the rules of the grammar specified for the regular expression object, modified according to the effects listed below for any bitmask elements set.

- enum [match_flag_type](#) : unsigned int {
[_S_default](#) , [_S_not_bol](#) , [_S_not_eol](#) , [_S_not_bow](#) ,
[_S_not_eow](#) , [_S_any](#) , [_S_not_null](#) , [_S_continuous](#) ,
[_S_prev_avail](#) , [_S_sed](#) , [_S_no_copy](#) , [_S_first_only](#) ,
[_S_match_flag_last](#) }
- constexpr [match_flag_type](#) match_default
- constexpr [match_flag_type](#) match_not_bol
- constexpr [match_flag_type](#) match_not_eol
- constexpr [match_flag_type](#) match_not_bow
- constexpr [match_flag_type](#) match_not_eow
- constexpr [match_flag_type](#) match_any
- constexpr [match_flag_type](#) match_not_null
- constexpr [match_flag_type](#) match_continuous
- constexpr [match_flag_type](#) match_prev_avail
- constexpr [match_flag_type](#) format_default
- constexpr [match_flag_type](#) format_sed
- constexpr [match_flag_type](#) format_no_copy
- constexpr [match_flag_type](#) format_first_only
- constexpr [match_flag_type](#) operator& (match_flag_type __a, match_flag_type __b)
- constexpr [match_flag_type](#) operator| (match_flag_type __a, match_flag_type __b)
- constexpr [match_flag_type](#) operator^ (match_flag_type __a, match_flag_type __b)
- constexpr [match_flag_type](#) operator~ (match_flag_type __a)
- constexpr [match_flag_type](#) & operator&= (match_flag_type &__a, match_flag_type __b)
- constexpr [match_flag_type](#) & operator|= (match_flag_type &__a, match_flag_type __b)
- constexpr [match_flag_type](#) & operator^= (match_flag_type &__a, match_flag_type __b)

5.3 Error Types

- enum [error_type](#) {
[_S_error_collate](#) , [_S_error_ctype](#) , [_S_error_escape](#) , [_S_error_backref](#) ,
[_S_error_brack](#) , [_S_error_paren](#) , [_S_error_brace](#) , [_S_error_badbrace](#) ,
[_S_error_range](#) , [_S_error_space](#) , [_S_error_badrepeat](#) , [_S_error_complexity](#) ,
[_S_error_stack](#) , [_S_null](#) , [_S_grammar](#) }
- constexpr [error_type](#) error_collate (_S_error_collate)
- constexpr [error_type](#) error_ctype (_S_error_ctype)
- constexpr [error_type](#) error_escape (_S_error_escape)
- constexpr [error_type](#) error_backref (_S_error_backref)
- constexpr [error_type](#) error_brack (_S_error_brack)
- constexpr [error_type](#) error_paren (_S_error_paren)
- constexpr [error_type](#) error_brace (_S_error_brace)
- constexpr [error_type](#) error_badbrace (_S_error_badbrace)
- constexpr [error_type](#) error_range (_S_error_range)

- constexpr [error_type error_space](#) ([_S_error_space](#))
- constexpr [error_type error_badrepeat](#) ([_S_error_badrepeat](#))
- constexpr [error_type error_complexity](#) ([_S_error_complexity](#))
- constexpr [error_type error_stack](#) ([_S_error_stack](#))

4.22.1 Detailed Description

ISO C++ 2011 namespace for options and flags used with `std::regex`.

4.22.2 Enumeration Type Documentation

error_type

```
enum std::regex\_constants::error\_type
```

The expression contained an invalid collating element name.

match_flag_type

```
enum std::regex\_constants::match\_flag\_type : unsigned int
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

syntax_option_type

```
enum std::regex\_constants::syntax\_option\_type : unsigned int
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

4.22.3 Function Documentation

error_backref()

```
constexpr error\_type std::regex\_constants::error\_backref (  
    \_S\_error\_backref ) [inline], [constexpr]
```

The expression contained an invalid back reference.

error_badbrace()

```
constexpr error\_type std::regex\_constants::error\_badbrace (  
    \_S\_error\_badbrace ) [inline], [constexpr]
```

The expression contained an invalid range in a `{}` expression.

error_badrepeat()

```
constexpr error\_type std::regex\_constants::error\_badrepeat (  
    \_S\_error\_badrepeat ) [inline], [constexpr]
```

One of `*?+{` was not preceded by a valid regular expression.

error_brace()

```
constexpr error\_type std::regex\_constants::error\_brace (  
    \_S\_error\_brace ) [inline], [constexpr]
```

The expression contained mismatched `{` and `}`

error_brack()

```
constexpr error_type std::regex_constants::error_brack (
    _S_error_brack ) [inline], [constexpr]
```

The expression contained mismatched [and].

error_collate()

```
constexpr error_type std::regex_constants::error_collate (
    _S_error_collate ) [inline], [constexpr]
```

The expression contained an invalid collating element name.

error_complexity()

```
constexpr error_type std::regex_constants::error_complexity (
    _S_error_complexity ) [inline], [constexpr]
```

The complexity of an attempted match against a regular expression exceeded a pre-set level.

error_ctype()

```
constexpr error_type std::regex_constants::error_ctype (
    _S_error_ctype ) [inline], [constexpr]
```

The expression contained an invalid character class name.

error_escape()

```
constexpr error_type std::regex_constants::error_escape (
    _S_error_escape ) [inline], [constexpr]
```

The expression contained an invalid escaped character, or a trailing escape.

error_paren()

```
constexpr error_type std::regex_constants::error_paren (
    _S_error_paren ) [inline], [constexpr]
```

The expression contained mismatched (and).

error_range()

```
constexpr error_type std::regex_constants::error_range (
    _S_error_range ) [inline], [constexpr]
```

The expression contained an invalid character range, such as [b-a] in most encodings.

error_space()

```
constexpr error_type std::regex_constants::error_space (
    _S_error_space ) [inline], [constexpr]
```

There was insufficient memory to convert the expression into a finite state machine.

error_stack()

```
constexpr error_type std::regex_constants::error_stack (
    _S_error_stack ) [inline], [constexpr]
```

There was insufficient memory to determine whether the regular expression could match the specified character sequence.

operator&() [1/2]

```
constexpr match_flag_type std::regex_constants::operator& (
    match_flag_type __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator&() [2/2]

```
constexpr syntax_option_type std::regex_constants::operator& (
    syntax_option_type __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator&=() [1/2]

```
constexpr match_flag_type & std::regex_constants::operator&= (
    match_flag_type & __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator&=() [2/2]

```
constexpr syntax_option_type & std::regex_constants::operator&= (
    syntax_option_type & __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator^() [1/2]

```
constexpr match_flag_type std::regex_constants::operator^ (
    match_flag_type __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator^() [2/2]

```
constexpr syntax_option_type std::regex_constants::operator^ (
    syntax_option_type __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator^=() [1/2]

```
constexpr match_flag_type & std::regex_constants::operator^= (
    match_flag_type & __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator^=() [2/2]

```
constexpr syntax_option_type & std::regex_constants::operator^= (
    syntax_option_type & __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator" |() [1/2]

```
constexpr match_flag_type std::regex_constants::operator| (
    match_flag_type __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator" |() [2/2]

```
constexpr syntax_option_type std::regex_constants::operator| (
    syntax_option_type __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator" |=() [1/2]

```
constexpr match_flag_type & std::regex_constants::operator|= (
    match_flag_type & __a,
    match_flag_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator" | =() [2/2]

```
constexpr syntax_option_type & std::regex_constants::operator|= (
    syntax_option_type & __a,
    syntax_option_type __b ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

operator~() [1/2]

```
constexpr match_flag_type std::regex_constants::operator~ (
    match_flag_type __a ) [inline], [constexpr]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

operator~() [2/2]

```
constexpr syntax_option_type std::regex_constants::operator~ (
    syntax_option_type __a ) [inline], [constexpr]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

4.22.4 Variable Documentation**__multiline**

```
constexpr syntax_option_type std::regex_constants::__multiline [inline], [constexpr]
```

Extension: Equivalent to `regex_constants::multiline` for C++11 and C++14.

__polynomial

```
constexpr syntax_option_type std::regex_constants::__polynomial [inline], [constexpr]
```

Extension: Ensure both space complexity of compiled regex and time complexity execution are not exponential. If specified in a regex with back-references, the exception `regex_constants::error_complexity` will be thrown.

awk

```
constexpr syntax_option_type std::regex_constants::awk [inline], [constexpr]
```

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility `awk` in IEEE Std 1003.1-2001. This option is identical to `syntax_option_type` `extended`, except that C-style escape sequences are supported. These sequences are: `\\`, `\a`, `\b`, `\f`, `\n`, `\r`, `\t`, `\v`, `\'`, `'`, and `\ddd` (where `ddd` is one, two, or three octal digits).

basic

```
constexpr syntax_option_type std::regex_constants::basic [inline], [constexpr]
```

Specifies that the grammar recognized by the regular expression engine is that used by POSIX basic regular expressions in IEEE Std 1003.1-2001, Portable Operating System Interface (POSIX), Base Definitions and Headers, Section 9,

Regular Expressions [IEEE, Information Technology – Portable Operating System Interface (POSIX), IEEE Standard 1003.1-2001].

collate

constexpr [syntax_option_type](#) std::regex_constants::collate [inline], [constexpr]

Specifies that character ranges of the form [a-b] should be locale sensitive.

ECMAScript

constexpr [syntax_option_type](#) std::regex_constants::ECMAScript [inline], [constexpr]

Specifies that the grammar recognized by the regular expression engine is that used by ECMAScript in ECMA-262 [Ecma International, ECMAScript Language Specification, Standard Ecma-262, third edition, 1999], as modified in section [28.13]. This grammar is similar to that defined in the PERL scripting language but extended with elements found in the POSIX regular expression grammar.

egrep

constexpr [syntax_option_type](#) std::regex_constants::egrep [inline], [constexpr]

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility grep when given the -E option in IEEE Std 1003.1-2001. This option is identical to [syntax_option_type](#) extended, except that newlines are treated as whitespace.

extended

constexpr [syntax_option_type](#) std::regex_constants::extended [inline], [constexpr]

Specifies that the grammar recognized by the regular expression engine is that used by POSIX extended regular expressions in IEEE Std 1003.1-2001, Portable Operating System Interface (POSIX), Base Definitions and Headers, Section 9, Regular Expressions.

format_default

constexpr [match_flag_type](#) std::regex_constants::format_default [inline], [constexpr]

When a regular expression match is to be replaced by a new string, the new string is constructed using the rules used by the ECMAScript replace function in ECMA- 262 [Ecma International, ECMAScript Language Specification, Standard Ecma-262, third edition, 1999], part 15.5.4.11 String.prototype.replace. In addition, during search and replace operations all non-overlapping occurrences of the regular expression are located and replaced, and sections of the input that did not match the expression are copied unchanged to the output string.

Format strings (from ECMA-262 [15.5.4.11]):

- \$\$ The dollar-sign itself (\$)
- \$& The matched substring.
- \$' The portion of *string* that precedes the matched substring. This would be `match_results::prefix()`.
- \$' The portion of *string* that follows the matched substring. This would be `match_results::suffix()`.
- \$n The nth capture, where n is in [1,9] and \$n is not followed by a decimal digit. If `n <= match_results::size()` and the nth capture is undefined, use the empty string instead. If `n > match_results::size()`, the result is implementation-defined.
- \$nn The nnth capture, where nn is a two-digit decimal number on [01, 99]. If `nn <= match_results::size()` and the nth capture is undefined, use the empty string instead. If `nn > match_results::size()`, the result is implementation-defined.

format_first_only

constexpr `match_flag_type` `std::regex_constants::format_first_only` `[inline]`, `[constexpr]`

When specified during a search and replace operation, only the first occurrence of the regular expression shall be replaced.

format_no_copy

constexpr `match_flag_type` `std::regex_constants::format_no_copy` `[inline]`, `[constexpr]`

During a search and replace operation, sections of the character container sequence being searched that do not match the regular expression shall not be copied to the output string.

format_sed

constexpr `match_flag_type` `std::regex_constants::format_sed` `[inline]`, `[constexpr]`

When a regular expression match is to be replaced by a new string, the new string is constructed using the rules used by the POSIX sed utility in IEEE Std 1003.1- 2001 [IEEE, Information Technology – Portable Operating System Interface (POSIX), IEEE Standard 1003.1-2001].

grep

constexpr `syntax_option_type` `std::regex_constants::grep` `[inline]`, `[constexpr]`

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility grep in IEEE Std 1003.1-2001. This option is identical to `syntax_option_type` basic, except that newlines are treated as whitespace.

icase

constexpr `syntax_option_type` `std::regex_constants::icase` `[inline]`, `[constexpr]`

Specifies that the matching of regular expressions against a character sequence shall be performed without regard to case.

match_any

constexpr `match_flag_type` `std::regex_constants::match_any` `[inline]`, `[constexpr]`

If more than one match is possible then any match is an acceptable result.

match_continuous

constexpr `match_flag_type` `std::regex_constants::match_continuous` `[inline]`, `[constexpr]`

The expression only matches a sub-sequence that begins at first .

match_default

constexpr `match_flag_type` `std::regex_constants::match_default` `[inline]`, `[constexpr]`

The default matching rules.

match_not_bol

constexpr `match_flag_type` `std::regex_constants::match_not_bol` `[inline]`, `[constexpr]`

The first character in the sequence [first, last) is treated as though it is not at the beginning of a line, so the character (^) in the regular expression shall not match [first, first).

match_not_bow

constexpr `match_flag_type` `std::regex_constants::match_not_bow` `[inline]`, `[constexpr]`

The expression \b is not matched against the sub-sequence [first,first).

match_not_eol

constexpr [match_flag_type](#) std::regex_constants::match_not_eol [inline], [constexpr]

The last character in the sequence [first, last) is treated as though it is not at the end of a line, so the character (\$) in the regular expression shall not match [last, last).

match_not_eow

constexpr [match_flag_type](#) std::regex_constants::match_not_eow [inline], [constexpr]

The expression \b should not be matched against the sub-sequence [last,last).

match_not_null

constexpr [match_flag_type](#) std::regex_constants::match_not_null [inline], [constexpr]

The expression does not match an empty sequence.

match_prev_avail

constexpr [match_flag_type](#) std::regex_constants::match_prev_avail [inline], [constexpr]

--first is a valid iterator position. When this flag is set then the flags match_not_bol and match_not_bow are ignored by the algorithms regex_match, regex_search, and regex_replace, and by the iterators regex_iterator and regex_token_iterator.

multiline

constexpr [syntax_option_type](#) std::regex_constants::multiline [inline], [constexpr]

Specifies that the ^ anchor matches at the beginning of a line, and the \$ anchor matches at the end of a line, not only at the beginning/end of the input. Valid for the ECMAScript syntax, ignored otherwise.

Since

C++17

nosubs

constexpr [syntax_option_type](#) std::regex_constants::nosubs [inline], [constexpr]

Specifies that when a regular expression is matched against a character container sequence, no sub-expression matches are to be stored in the supplied match_results structure.

optimize

constexpr [syntax_option_type](#) std::regex_constants::optimize [inline], [constexpr]

Specifies that the regular expression engine should pay more attention to the speed with which regular expressions are matched, and less to the speed with which regular expression objects are constructed. Otherwise it has no detectable effect on the program output.

4.23 std::rel_ops Namespace Reference**Functions**

- template<class _Tp >
bool [operator!=](#) (const _Tp &__x, const _Tp &__y)
- template<class _Tp >
bool [operator<=](#) (const _Tp &__x, const _Tp &__y)
- template<class _Tp >
bool [operator>](#) (const _Tp &__x, const _Tp &__y)

- `template<class _Tp >`
`bool operator>= (const _Tp &__x, const _Tp &__y)`

4.23.1 Detailed Description

The generated relational operators are sequestered here.

Libstdc++ headers must not use the contents of `rel_ops`. User code should also avoid them, because unconstrained function templates are too greedy and can easily cause ambiguities.

C++20 default comparisons are a better solution.

4.23.2 Function Documentation

operator"!=()

```
template<class _Tp >
bool std::rel_ops::operator!= (
    const _Tp & __x,
    const _Tp & __y ) [inline]
```

Defines != for arbitrary types, in terms of ==.

Parameters

<code>__x</code>	A thing.
<code>__y</code>	Another thing.

Returns

`__x != __y`

This function uses == to determine its result.

operator<=()

```
template<class _Tp >
bool std::rel_ops::operator<= (
    const _Tp & __x,
    const _Tp & __y ) [inline]
```

Defines <= for arbitrary types, in terms of <.

Parameters

<code>__x</code>	A thing.
<code>__y</code>	Another thing.

Returns

`__x <= __y`

This function uses < to determine its result.

operator>()

```
template<class _Tp >
bool std::rel_ops::operator> (
    const _Tp & __x,
    const _Tp & __y ) [inline]
```

Defines > for arbitrary types, in terms of <.

Parameters

$\begin{smallmatrix} \leftarrow \\ _x \end{smallmatrix}$	A thing.
$\begin{smallmatrix} \leftarrow \\ _y \end{smallmatrix}$	Another thing.

Returns

`__x > __y`

This function uses < to determine its result.

operator>=()

```
template<class _Tp >
bool std::rel_ops::operator>= (
    const _Tp & __x,
    const _Tp & __y ) [inline]
```

Defines >= for arbitrary types, in terms of <.

Parameters

$\begin{smallmatrix} \leftarrow \\ _x \end{smallmatrix}$	A thing.
$\begin{smallmatrix} \leftarrow \\ _y \end{smallmatrix}$	Another thing.

Returns

`__x >= __y`

This function uses < to determine its result.

4.24 std::this_thread Namespace Reference**Functions**

- [thread::id get_id \(\)](#) noexcept
- `template<typename _Rep , typename _Period >`
void [sleep_for](#) (const [chrono::duration](#)< _Rep, _Period > &__rtime)

- `template<typename _Clock, typename _Duration >`
`void sleep_until (const chrono::time_point< _Clock, _Duration > &__atime)`
- `void yield () noexcept`

4.24.1 Detailed Description

ISO C++ 2011 namespace for interacting with the current thread.

C++11 30.3.2 [thread.thread.this] Namespace `this_thread`.

4.24.2 Function Documentation

`get_id()`

`thread::id std::this_thread::get_id () [inline], [noexcept]`

The unique identifier of the current thread.

`sleep_for()`

`template<typename _Rep, typename _Period >`
`void std::this_thread::sleep_for (`
`const chrono::duration< _Rep, _Period > &__rtime) [inline]`

`this_thread::sleep_for`

Referenced by [sleep_until\(\)](#).

`sleep_until()`

`template<typename _Clock, typename _Duration >`
`void std::this_thread::sleep_until (`
`const chrono::time_point< _Clock, _Duration > &__atime) [inline]`

`this_thread::sleep_until`

References [sleep_for\(\)](#).

`yield()`

`void std::this_thread::yield () [inline], [noexcept]`

Allow the implementation to schedule a different thread.

4.25 std::tr1 Namespace Reference

Namespaces

- namespace [__detail](#)

Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<typename _Tpx, typename _Tpy >`
`__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type beta (_Tpx __x, _Tpy __y)`

- float **betaf** (float __x, float __y)
- long double **betal** (long double __x, long double __y)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **comp_ellint_1** (_Tp __k)
- float **comp_ellint_1f** (float __k)
- long double **comp_ellint_1l** (long double __k)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **comp_ellint_2** (_Tp __k)
- float **comp_ellint_2f** (float __k)
- long double **comp_ellint_2l** (long double __k)
- template<typename _Tp, typename _Tpn >
__gnu_cxx::__promote_2< _Tp, _Tpn >::__type **comp_ellint_3** (_Tp __k, _Tpn __nu)
- float **comp_ellint_3f** (float __k, float __nu)
- long double **comp_ellint_3l** (long double __k, long double __nu)
- template<typename _Tpa, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type **conf_hypergf** (_Tpa __a, _Tpc __c, _Tp __x)
- float **conf_hypergfl** (float __a, float __c, float __x)
- long double **conf_hypergll** (long double __a, long double __c, long double __x)
- template<typename _Tp >
std::complex< typename __gnu_cxx::__promote< _Tp >::__type > **conj** (_Tp __x)
- template<typename _Tp >
std::complex< _Tp > **conj** (const **std::complex**< _Tp > &__z)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_i** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_if** (float __nu, float __x)
- long double **cyl_bessel_il** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_j** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_jf** (float __nu, float __x)
- long double **cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_kf** (float __nu, float __x)
- long double **cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **cyl_neumannf** (float __nu, float __x)
- long double **cyl_neumannl** (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **ellint_1** (_Tp __k, _Tpp __phi)
- float **ellint_1f** (float __k, float __phi)
- long double **ellint_1l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **ellint_2** (_Tp __k, _Tpp __phi)
- float **ellint_2f** (float __k, float __phi)
- long double **ellint_2l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type **ellint_3** (_Tp __k, _Tpn __nu, _Tpp __phi)
- float **ellint_3f** (float __k, float __nu, float __phi)
- long double **ellint_3l** (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **expint** (_Tp __x)

- float **expintf** (float __x)
- long double **expintl** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **fabs** (_Tp __x)
- template<typename _Tp >
[std::complex](#)< _Tp > **fabs** (const [std::complex](#)< _Tp > &__z)
- float **fabs** (float __x)
- long double **fabs** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **hermite** (unsigned int __n, _Tp __x)
- float **hermitef** (unsigned int __n, float __x)
- long double **hermitel** (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type **hyperg** (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)
- float **hypergfl** (float __a, float __b, float __c, float __x)
- long double **hypergl** (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **laguerre** (unsigned int __n, _Tp __x)
- float **laguerref** (unsigned int __n, float __x)
- long double **laguerrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **legendre** (unsigned int __n, _Tp __x)
- float **legendref** (unsigned int __n, float __x)
- long double **legendrel** (unsigned int __n, long double __x)
- template<typename _Tp, typename _Up >
[std::complex](#)< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > **polar** (const _Tp &__rho, const _Up &__theta)
- template<typename _Tp, typename _Up >
__gnu_cxx::__promote_2< _Tp, _Up >::__type **pow** (_Tp __x, _Up __y)
- template<typename _Tp >
[std::complex](#)< _Tp > **pow** (const _Tp &__x, const [std::complex](#)< _Tp > &__y)
- template<typename _Tp, typename _Up >
[std::complex](#)< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > **pow** (const _Tp &__x, const [std::complex](#)< _Up > &__y)
- template<typename _Tp >
[std::complex](#)< _Tp > **pow** (const [std::complex](#)< _Tp > &__x, const _Tp &__y)
- template<typename _Tp, typename _Up >
[std::complex](#)< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > **pow** (const [std::complex](#)< _Tp > &__x, const _Up &__y)
- template<typename _Tp >
[std::complex](#)< _Tp > **pow** (const [std::complex](#)< _Tp > &__x, const [std::complex](#)< _Tp > &__y)
- template<typename _Tp, typename _Up >
[std::complex](#)< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > **pow** (const [std::complex](#)< _Tp > &__x, const [std::complex](#)< _Up > &__y)
- float **pow** (float __x, float __y)
- long double **pow** (long double __x, long double __y)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **riemann_zeta** (_Tp __x)
- float **riemann_zetaf** (float __x)
- long double **riemann_zetal** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **sph_bessel** (unsigned int __n, _Tp __x)

- float **sph_besself** (unsigned int __n, float __x)
- long double **sph_bessell** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **sph_legendre** (unsigned int __l, unsigned int __m, _Tp __theta)
- float **sph_legendref** (unsigned int __l, unsigned int __m, float __theta)
- long double **sph_legendrel** (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **sph_neumann** (unsigned int __n, _Tp __x)
- float **sph_neumannf** (unsigned int __n, float __x)
- long double **sph_neumannl** (unsigned int __n, long double __x)

4.25.1 Detailed Description

ISO C++ TR1 entities toplevel namespace is `std::tr1`.

4.26 `std::tr1::__detail` Namespace Reference

4.26.1 Detailed Description

Implementation details not part of the namespace `std::tr1` interface.

4.27 `std::tr2` Namespace Reference

Namespaces

- namespace [`__detail`](#)

Classes

- struct [`__dynamic_bitset_base`](#)
- struct [`__reflection_typelist`](#)
- struct [`__reflection_typelist< _First, _Rest... >`](#)
- struct [`__reflection_typelist<>`](#)
- struct [`bases`](#)
- class [`bool_set`](#)
- struct [`direct_bases`](#)
- class [`dynamic_bitset`](#)

Functions

- bool **certainly** ([`bool_set`](#) __b)
- bool **contains** ([`bool_set`](#) __s, [`bool_set`](#) __t)
- bool **equals** ([`bool_set`](#) __s, [`bool_set`](#) __t)
- bool **is_emptyset** ([`bool_set`](#) __b)
- bool **is_indeterminate** ([`bool_set`](#) __b)
- bool **is_singleton** ([`bool_set`](#) __b)
- [`bool_set`](#) **operator!=** ([`bool`](#) __s, [`bool_set`](#) __t)
- [`bool_set`](#) **operator!=** ([`bool_set`](#) __s, [`bool`](#) __t)
- [`bool_set`](#) **operator!=** ([`bool_set`](#) __s, [`bool_set`](#) __t)
- [`bool_set`](#) **operator&** ([`bool`](#) __s, [`bool_set`](#) __t)
- [`bool_set`](#) **operator&** ([`bool_set`](#) __s, [`bool`](#) __t)
- template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >
[`std::basic_ostream< _CharT, _Traits >`](#) & **operator<<** ([`std::basic_ostream< _CharT, _Traits >`](#) &__os, const [`dynamic_bitset< _WordT, _Alloc >`](#) &__x)

- `bool_set operator==` (`bool __s`, `bool_set __t`)
- `bool_set operator==` (`bool_set __s`, `bool __t`)
- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & operator>>` (`std::basic_istream< _CharT, _Traits > &__is`,
`dynamic_bitset< _WordT, _Alloc > &__x`)
- `bool_set operator^` (`bool __s`, `bool_set __t`)
- `bool_set operator^` (`bool_set __s`, `bool __t`)
- `bool_set operator|` (`bool __s`, `bool_set __t`)
- `bool_set operator|` (`bool_set __s`, `bool __t`)
- `bool possibly` (`bool_set __b`)
- `bool_set set_complement` (`bool_set __b`)
- `bool_set set_intersection` (`bool __s`, `bool_set __t`)
- `bool_set set_intersection` (`bool_set __s`, `bool __t`)
- `bool_set set_intersection` (`bool_set __s`, `bool_set __t`)
- `bool_set set_union` (`bool __s`, `bool_set __t`)
- `bool_set set_union` (`bool_set __s`, `bool __t`)
- `bool_set set_union` (`bool_set __s`, `bool_set __t`)

- `template<typename _WordT, typename _Alloc >`
`bool operator!=` (`const dynamic_bitset< _WordT, _Alloc > &__lhs`, `const dynamic_bitset< _WordT, _Alloc >`
`&__rhs`)
- `template<typename _WordT, typename _Alloc >`
`bool operator<=` (`const dynamic_bitset< _WordT, _Alloc > &__lhs`, `const dynamic_bitset< _WordT, _Alloc >`
`&__rhs`)
- `template<typename _WordT, typename _Alloc >`
`bool operator>` (`const dynamic_bitset< _WordT, _Alloc > &__lhs`, `const dynamic_bitset< _WordT, _Alloc >`
`&__rhs`)
- `template<typename _WordT, typename _Alloc >`
`bool operator>=` (`const dynamic_bitset< _WordT, _Alloc > &__lhs`, `const dynamic_bitset< _WordT, _Alloc >`
`&__rhs`)

- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator&` (`const dynamic_bitset< _WordT, _Alloc > &__x`, `const`
`dynamic_bitset< _WordT, _Alloc > &__y`)
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator|` (`const dynamic_bitset< _WordT, _Alloc > &__x`, `const`
`dynamic_bitset< _WordT, _Alloc > &__y`)
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator^` (`const dynamic_bitset< _WordT, _Alloc > &__x`, `const`
`dynamic_bitset< _WordT, _Alloc > &__y`)
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator-` (`const dynamic_bitset< _WordT, _Alloc > &__x`, `const`
`dynamic_bitset< _WordT, _Alloc > &__y`)

4.27.1 Detailed Description

Namespace for non-standard "TR2" extensions.

4.28 std::tr2::__detail Namespace Reference

4.28.1 Detailed Description

Implementation details not part of the namespace `std::tr2` interface.

5 Concept Documentation

5.1 `std::assignable_from` Concept Reference

5.1.1 Concept definition

```
template<typename _Lhs, typename _Rhs>
concept std::assignable_from = is_lvalue_reference_v<_Lhs>
    && common_reference_with<__detail::__cref<_Lhs>, __detail::__cref<_Rhs>
    && requires(_Lhs __lhs, _Rhs&& __rhs) {
        { __lhs = static_cast<_Rhs&&>(__rhs) } -> same_as<_Lhs>;
    }
```

5.1.2 Detailed Description

[concept.assignable], concept `assignable_from`

5.2 `std::common_reference_with` Concept Reference

5.2.1 Concept definition

```
template<typename _Tp, typename _Up>
concept std::common_reference_with = same_as<common_reference_t<_Tp, _Up>, common_reference_t<_Up, _Tp>
    && convertible_to<_Tp, common_reference_t<_Tp, _Up>>
    && convertible_to<_Up, common_reference_t<_Tp, _Up>>
```

5.2.2 Detailed Description

[concept.commonref], concept `common_reference_with`

5.3 `std::common_with` Concept Reference

5.3.1 Concept definition

```
template<typename _Tp, typename _Up>
concept std::common_with = same_as<common_type_t<_Tp, _Up>, common_type_t<_Up, _Tp>
    && requires {
        static_cast<common_type_t<_Tp, _Up>>(std::declval<_Tp>());
        static_cast<common_type_t<_Tp, _Up>>(std::declval<_Up>());
    }
    && common_reference_with<add_lvalue_reference_t<const _Tp>,
        add_lvalue_reference_t<const _Up>>
    && common_reference_with<add_lvalue_reference_t<common_type_t<_Tp, _Up>>,
        common_reference_t<
            add_lvalue_reference_t<const _Tp>,
            add_lvalue_reference_t<const _Up>>>
```

5.3.2 Detailed Description

[concept.common], concept `common_with`

5.4 `std::constructible_from` Concept Reference

5.4.1 Concept definition

```
template<typename _Tp, typename... _Args>
concept std::constructible_from = destructible<_Tp> && is_constructible_v<_Tp, _Args...>
```

5.4.2 Detailed Description

[concept.constructible], concept `constructible_from`

5.5 std::convertible_to Concept Reference

5.5.1 Concept definition

```
template<typename _From, typename _To>
concept std::convertible_to = is_convertible_v<_From, _To>
    && requires { static_cast<_To>(std::declval<_From>()); }
```

5.5.2 Detailed Description

[concept.convertible], concept convertible_to

5.6 std::copy_constructible Concept Reference

5.6.1 Concept definition

```
template<typename _Tp>
concept std::copy_constructible = move_constructible<_Tp>
    && constructible_from<_Tp, _Tp&> && convertible_to<_Tp&, _Tp>
    && constructible_from<_Tp, const _Tp&> && convertible_to<const _Tp&, _Tp>
    && constructible_from<_Tp, const _Tp> && convertible_to<const _Tp, _Tp>
```

5.6.2 Detailed Description

[concept.copyconstructible], concept copy_constructible

5.7 std::default_initializable Concept Reference

5.7.1 Concept definition

```
template<typename _Tp>
concept std::default_initializable = constructible_from<_Tp>
    && requires
    {
        _Tp{};
        (void) ::new _Tp;
    }
```

5.7.2 Detailed Description

[concept.defaultinitializable], concept default_initializable

5.8 std::derived_from Concept Reference

5.8.1 Concept definition

```
template<typename _Derived, typename _Base>
concept std::derived_from = __is_base_of(_Base, _Derived)
    && is_convertible_v<const volatile _Derived*, const volatile _Base*>
```

5.8.2 Detailed Description

[concept.derived], concept derived_from

5.9 std::destructible Concept Reference

5.9.1 Concept definition

```
template<typename _Tp>
concept std::destructible = is_nothrow_destructible_v<_Tp>
```

5.9.2 Detailed Description

[concept.destructible], concept destructible

5.10 std::equivalence_relation Concept Reference

5.10.1 Concept definition

```
template<typename _Rel, typename _Tp, typename _Up>
concept std::equivalence_relation = relation<_Rel, _Tp, _Up>
```

5.10.2 Detailed Description

[concept.equiv], concept equivalence_relation

5.11 std::indirectly_comparable Concept Reference

```
#include <iterator_concepts.h>
```

5.11.1 Concept definition

```
template<typename _I1, typename _I2, typename _Rel, typename _P1 identity, typename _P2 identity>
concept std::indirectly_comparable = indirect_binary_predicate<_Rel, projected<_I1, _P1>,
    projected<_I2, _P2>>
```

5.11.2 Detailed Description

[alg.req.ind.cmp], concept indirectly_comparable

5.12 std::indirectly_copyable Concept Reference

```
#include <iterator_concepts.h>
```

5.12.1 Concept definition

```
template<typename _In, typename _Out>
concept std::indirectly_copyable = indirectly_readable<_In>
    && indirectly_writable<_Out, iter_reference_t<_In>>
```

5.12.2 Detailed Description

[alg.req.ind.copy], concept indirectly_copyable

5.13 std::indirectly_movable Concept Reference

```
#include <iterator_concepts.h>
```

5.13.1 Concept definition

```
template<typename _In, typename _Out>
concept std::indirectly_movable = indirectly_readable<_In>
    && indirectly_writable<_Out, iter_rvalue_reference_t<_In>>
```

5.13.2 Detailed Description

[alg.req.ind.move], concept indirectly_movable

5.14 std::indirectly_readable Concept Reference

```
#include <iterator_concepts.h>
```

5.14.1 Concept definition

```
template<typename _In>
concept std::indirectly_readable = __detail::__indirectly_readable_impl<remove_cvref_t<_In>
```

5.14.2 Detailed Description

Requirements for types that are readable by applying operator*.

5.15 std::indirectly_swappable Concept Reference

```
#include <iterator_concepts.h>
```

5.15.1 Concept definition

```
template<typename _I1, typename _I2 _I1>
concept std::indirectly_swappable = indirectly_readable<_I1> && indirectly_readable<_I2>
    && requires(const _I1 __i1, const _I2 __i2)
    {
        ranges::iter_swap(__i1, __i1);
        ranges::iter_swap(__i2, __i2);
        ranges::iter_swap(__i1, __i2);
        ranges::iter_swap(__i2, __i1);
    }
```

5.15.2 Detailed Description

[alg.req.ind.swap], concept indirectly_swappable

5.16 std::indirectly_writable Concept Reference

```
#include <iterator_concepts.h>
```

5.16.1 Concept definition

```
template<typename _Out, typename _Tp>
concept std::indirectly_writable = requires(_Out&& __o, _Tp&& __t)
{
    *__o = std::forward<_Tp>(__t);
    std::forward<_Out>(__o) = std::forward<_Tp>(__t);
    const_cast<const iter_reference_t<_Out>&&>(*__o)
        = std::forward<_Tp>(__t);
    const_cast<const iter_reference_t<_Out>&&>(*std::forward<_Out>(__o))
        = std::forward<_Tp>(__t);
}
```

5.16.2 Detailed Description

Requirements for writing a value into an iterator's referenced object.

5.17 std::invocable Concept Reference

5.17.1 Concept definition

```
template<typename _Fn, typename... _Args>
concept std::invocable = is_invocable_v<_Fn, _Args...>
```

5.17.2 Detailed Description

[concept.invocable], concept invocable

5.18 std::mergeable Concept Reference

```
#include <iterator_concepts.h>
```

5.18.1 Concept definition

```
template<typename _I1, typename _I2, typename _Out, typename _Rel ranges::less, typename _P1 identity, typename
_P2 identity>
concept std::mergeable = input_iterator<_I1> && input_iterator<_I2>
&& weakly_incrementable<_Out> && indirectly_copyable<_I1, _Out>
&& indirectly_copyable<_I2, _Out>
&& indirect_strict_weak_order<_Rel, projected<_I1, _P1>,
projected<_I2, _P2>
```

5.18.2 Detailed Description

[alg.req.mergeable], concept mergeable

5.19 std::move_constructible Concept Reference

5.19.1 Concept definition

```
template<typename _Tp>
concept std::move_constructible = constructible_from<_Tp, _Tp> && convertible_to<_Tp, _Tp>
```

5.19.2 Detailed Description

[concept.moveconstructible], concept move_constructible

5.20 std::permutable Concept Reference

```
#include <iterator_concepts.h>
```

5.20.1 Concept definition

```
template<typename _Iter>
concept std::permutable = forward_iterator<_Iter>
&& indirectly_movable_storable<_Iter, _Iter>
&& indirectly_swappable<_Iter, _Iter>
```

5.20.2 Detailed Description

[alg.req.permutable], concept permutable

5.21 std::predicate Concept Reference

5.21.1 Concept definition

```
template<typename _Fn, typename... _Args>
concept std::predicate = regular_invocable<_Fn, _Args...>
&& __detail::__boolean_testable<invoke_result_t<_Fn, _Args...>
```

5.21.2 Detailed Description

[concept.predicate], concept predicate

5.22 std::ranges::bidirectional_range Concept Reference

```
#include <ranges_base.h>
```

5.22.1 Concept definition

```
template<typename _Tp>
concept std::ranges::bidirectional_range = forward_range<_Tp> && bidirectional_iterator<iterator_t<_Tp>
```


5.22.2 Detailed Description

A range for which `ranges::begin` returns a bidirectional iterator.

5.23 `std::ranges::borrowed_range` Concept Reference

```
#include <ranges_base.h>
```

5.23.1 Concept definition

```
template<typename _Tp>
concept std::ranges::borrowed_range = range<_Tp> && __detail::__maybe_borrowed_range<_Tp>
```

5.23.2 Detailed Description

[range.range] The `borrowed_range` concept.

5.24 `std::ranges::common_range` Concept Reference

```
#include <ranges_base.h>
```

5.24.1 Concept definition

```
template<typename _Tp>
concept std::ranges::common_range = range<_Tp> && same_as<iterator_t<_Tp>, sentinel_t<_Tp>>
```

5.24.2 Detailed Description

A range for which `ranges::begin` and `ranges::end` return the same type.

5.25 `std::ranges::contiguous_range` Concept Reference

```
#include <ranges_base.h>
```

5.25.1 Concept definition

```
template<typename _Tp>
concept std::ranges::contiguous_range = random_access_range<_Tp> && contiguous_iterator<iterator_t<_Tp>>
    && requires(_Tp& __t)
    {
        { ranges::data(__t) } -> same_as<add_pointer_t<range_reference_t<_Tp>>;
    }
```

5.25.2 Detailed Description

A range for which `ranges::begin` returns a contiguous iterator.

5.26 `std::ranges::forward_range` Concept Reference

```
#include <ranges_base.h>
```

5.26.1 Concept definition

```
template<typename _Tp>
concept std::ranges::forward_range = input_range<_Tp> && forward_iterator<iterator_t<_Tp>>
```

5.26.2 Detailed Description

A range for which `ranges::begin` returns a forward iterator.

5.27 std::ranges::input_range Concept Reference

```
#include <ranges_base.h>
```

5.27.1 Concept definition

```
template<typename _Tp>
concept std::ranges::input_range = range<_Tp> && input_iterator<iterator_t<_Tp>>
```

5.27.2 Detailed Description

A range for which ranges::begin returns an input iterator.

5.28 std::ranges::output_range Concept Reference

```
#include <ranges_base.h>
```

5.28.1 Concept definition

```
template<typename _Range, typename _Tp>
concept std::ranges::output_range = range<_Range> && output_iterator<iterator_t<_Range>, _Tp>
```

5.28.2 Detailed Description

A range for which ranges::begin returns an output iterator.

5.29 std::ranges::random_access_range Concept Reference

```
#include <ranges_base.h>
```

5.29.1 Concept definition

```
template<typename _Tp>
concept std::ranges::random_access_range = bidirectional_range<_Tp> && random_access_iterator<iterator_t<_Tp>>
```

5.29.2 Detailed Description

A range for which ranges::begin returns a random access iterator.

5.30 std::ranges::range Concept Reference

```
#include <ranges_base.h>
```

5.30.1 Concept definition

```
template<typename _Tp>
concept std::ranges::range = requires(_Tp& __t)
{
    ranges::begin(__t);
    ranges::end(__t);
}
```

5.30.2 Detailed Description

[range.range] The range concept.

5.31 std::ranges::sized_range Concept Reference

```
#include <ranges_base.h>
```

5.31.1 Concept definition

```
template<typename _Tp>
concept std::ranges::sized_range = range<_Tp>
    && requires(_Tp& __t) { ranges::size(__t); }
```

5.31.2 Detailed Description

[range.sized] The sized_range concept.

5.32 std::ranges::view Concept Reference

```
#include <ranges_base.h>
```

5.32.1 Concept definition

```
template<typename _Tp>
concept std::ranges::view = range<_Tp> && movable<_Tp> && enable_view<_Tp>
```

5.32.2 Detailed Description

[range.view] The ranges::view concept.

5.33 std::ranges::viewable_range Concept Reference

```
#include <ranges_base.h>
```

5.33.1 Concept definition

```
template<typename _Tp>
concept std::ranges::viewable_range = range<_Tp>
    && ((view<remove_cvref_t<_Tp>> && constructible_from<remove_cvref_t<_Tp>, _Tp>)
        || (!view<remove_cvref_t<_Tp>>
            && (is_lvalue_reference_v<_Tp>
                || (movable<remove_reference_t<_Tp>>
                    && !__detail::__is_initializer_list<remove_cvref_t<_Tp>>))))
```

5.33.2 Detailed Description

A range which can be safely converted to a view.

5.34 std::regular_invocable Concept Reference

5.34.1 Concept definition

```
template<typename _Fn, typename... _Args>
concept std::regular_invocable = invocable<_Fn, _Args...>
```

5.34.2 Detailed Description

[concept.regularinvocable], concept regular_invocable

5.35 std::relation Concept Reference

5.35.1 Concept definition

```
template<typename _Rel, typename _Tp, typename _Up>
concept std::relation = predicate<_Rel, _Tp, _Tp> && predicate<_Rel, _Up, _Up>
    && predicate<_Rel, _Tp, _Up> && predicate<_Rel, _Up, _Tp>
```

5.35.2 Detailed Description

[concept.relation], concept relation

5.36 std::same_as Concept Reference

5.36.1 Concept definition

```
template<typename _Tp, typename _Up>
concept std::same_as = __detail::__same_as<_Tp, _Up> && __detail::__same_as<_Up, _Tp>
```

5.36.2 Detailed Description

[concept.same], concept same_as

5.37 std::sortable Concept Reference

```
#include <iterator_concepts.h>
```

5.37.1 Concept definition

```
template<typename _Iter, typename _Rel ranges::less, typename _Proj identity>
concept std::sortable = permutable<_Iter>
    && indirect_strict_weak_order<_Rel, projected<_Iter, _Proj>
```

5.37.2 Detailed Description

[alg.req.sortable], concept sortable

5.38 std::strict_weak_order Concept Reference

5.38.1 Concept definition

```
template<typename _Rel, typename _Tp, typename _Up>
concept std::strict_weak_order = relation<_Rel, _Tp, _Up>
```

5.38.2 Detailed Description

[concept.strictweakorder], concept strict_weak_order

5.39 std::uniform_random_bit_generator Concept Reference

```
#include <uniform_int_dist.h>
```

5.39.1 Concept definition

```
template<typename _Gen>
concept std::uniform_random_bit_generator = invocable<_Gen&> && unsigned_integral<invoke_result_t<_Gen&>
    && requires
    {
        { _Gen::min() } -> same_as<invoke_result_t<_Gen&>>;
        { _Gen::max() } -> same_as<invoke_result_t<_Gen&>>;
        requires bool_constant<(_Gen::min() < _Gen::max())>::value;
    }
```

5.39.2 Detailed Description

Requirements for a uniform random bit generator.

5.40 std::weakly_incrementable Concept Reference

```
#include <iterator_concepts.h>
```

5.40.1 Concept definition

```
template<typename _Iter>
concept std::weakly_incrementable = movable<_Iter>
    && requires(_Iter __i)
    {
        typename iter_difference_t<_Iter>;
        requires __detail::__is_signed_integer_like<iter_difference_t<_Iter>;
        { ++__i } -> same_as<_Iter&>;
        __i++;
    }
```

5.40.2 Detailed Description

Requirements on types that can be incremented with ++.

6 Class Documentation

6.1 `__gnu_parallel::__accumulate_binop_reduct<_BinOp >` Struct Template Reference

```
#include <for_each_selectors.h>
```

Public Member Functions

- `__accumulate_binop_reduct` (`_BinOp &__b`)
- `template<typename _Result, typename _Addend >`
`_Result operator()` (`const _Result &__x, const _Addend &__y`)

Public Attributes

- `_BinOp &__binop`

6.1.1 Detailed Description

```
template<typename _BinOp>
struct __gnu_parallel::__accumulate_binop_reduct<_BinOp >
```

General reduction, using a binary operator.

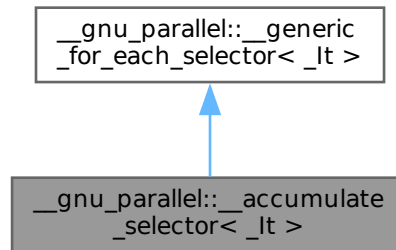
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.2 `__gnu_parallel::__accumulate_selector<_It >` Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__accumulate_selector<_It>`:



Public Member Functions

- `template<typename _Op>`
`std::iterator_traits<_It>::value_type operator() (_Op __o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.2.1 Detailed Description

`template<typename _It>`
`struct __gnu_parallel::__accumulate_selector<_It>`

`std::accumulate()` selector.

6.2.2 Member Function Documentation

`operator()()`

```

template<typename _It>
template<typename _Op>
std::iterator_traits<_It>::value_type __gnu_parallel::__accumulate_selector<_It>::operator()
(
    _Op __o,
    _It __i ) [inline]

```

Functor execution.

Parameters

<code>__o</code>	Operator (unused).
<code>__i</code>	iterator referencing object.

Returns

The current value.

6.2.3 Member Data Documentation**`_M_finish_iterator`**

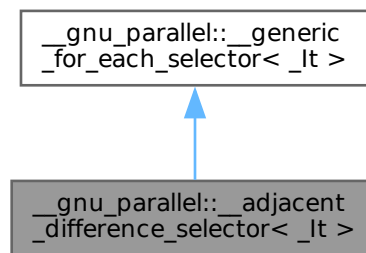
```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
The documentation for this struct was generated from the following file:
```

- [for_each_selectors.h](#)

6.3 `__gnu_parallel::__adjacent_difference_selector<_It>` Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__adjacent_difference_selector<_It>`:

**Public Member Functions**

- `template<typename _Op > bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.3.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__adjacent_difference_selector<_It>
```

Selector that returns the difference between two adjacent `__elements`.

6.3.2 Member Data Documentation**`_M_finish_iterator`**

```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
```

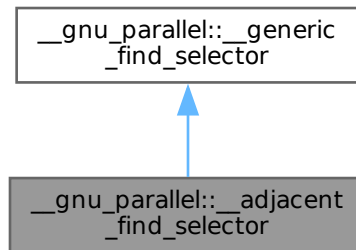
_Iterator on last element processed; needed for some algorithms (e. g. std::transform()).
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.4 __gnu_parallel::__adjacent_find_selector Struct Reference

```
#include <find_selectors.h>
```

Inheritance diagram for __gnu_parallel::__adjacent_find_selector:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair< _RAIter1, _RAIter2 > _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator() (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

6.4.1 Detailed Description

Test predicate on two adjacent elements.

6.4.2 Member Function Documentation

[_M_sequential_algorithm\(\)](#)

```

template<typename _RAIter1, typename _RAIter2, typename _Pred >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__adjacent_find_selector::_M_sequential_algorithm
(
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred ) [inline]
  
```

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.

Parameters

<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

operator>()

```
template<typename _RAIter1 , typename _RAIter2 , typename _Pred >
bool __gnu_parallel::__adjacent_find_selector::operator() (
    _RAIter1 __i1,
    _RAIter2 __i2,
    _Pred __pred ) [inline]
```

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

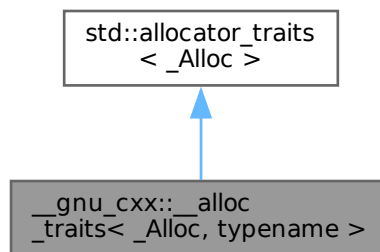
The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

6.5 __gnu_cxx::__alloc_traits<_Alloc, typename > Struct Template Reference

```
#include <alloc_traits.h>
```

Inheritance diagram for `__gnu_cxx::__alloc_traits<_Alloc, typename >`:

**Public Types**

- typedef [std::allocator_traits<_Alloc >](#) **_Base_type**
- typedef `_Alloc` **allocator_type**
- typedef [_Base_type::const_pointer](#) **const_pointer**
- typedef `const value_type &` **const_reference**
- using [const_void_pointer](#) = `typename _Ptr< __cv_pointer, const void >::type`

- typedef `_Base_type::difference_type` **difference_type**
- using `is_always_equal` = `__detected_or_t< typename is_empty< _Alloc >::type, __equal, _Alloc >`
- typedef `_Base_type::pointer` **pointer**
- using `propagate_on_container_copy_assignment` = `__detected_or_t< false_type, __pocca, _Alloc >`
- using `propagate_on_container_move_assignment` = `__detected_or_t< false_type, __pocma, _Alloc >`
- using `propagate_on_container_swap` = `__detected_or_t< false_type, __pocs, _Alloc >`
- template<typename `_Tp` >
using `rebind_alloc` = `__alloc_rebind< _Alloc, _Tp >`
- template<typename `_Tp` >
using `rebind_traits` = `allocator_traits< rebind_alloc< _Tp > >`
- typedef value_type & **reference**
- typedef `_Base_type::size_type` **size_type**
- typedef `_Base_type::value_type` **value_type**
- using `void_pointer` = `typename _Ptr< __v_pointer, void >::type`

Static Public Member Functions

- static constexpr bool `_S_always_equal` ()
- static constexpr bool `_S_nothrow_move` ()
- static constexpr void `_S_on_swap` (_Alloc &__a, _Alloc &__b)
- static constexpr bool `_S_propagate_on_copy_assign` ()
- static constexpr bool `_S_propagate_on_move_assign` ()
- static constexpr bool `_S_propagate_on_swap` ()
- static constexpr _Alloc `_S_select_on_copy` (const _Alloc &__a)
- static constexpr pointer `allocate` (_Alloc &__a, size_type __n)
- static constexpr pointer `allocate` (_Alloc &__a, size_type __n)
- static constexpr pointer `allocate` (_Alloc &__a, size_type __n, const_void_pointer __hint)
- static constexpr pointer `allocate` (_Alloc &__a, size_type __n, const_void_pointer __hint)
- template<typename `_Ptr`, typename... `_Args`>
static constexpr std::enable_if_t< __is_custom_pointer< _Ptr >::value > **construct** (_Alloc &__a, _Ptr __p, _Args &&... __args) noexcept(noexcept(_Base_type::construct(__a, std::to_address(__p), std::forward<_Args>(__args)...)))
- template<typename `_Tp`, typename... `_Args`>
static constexpr auto **construct** (_Alloc &__a, _Tp *__p, _Args &&... __args) noexcept(noexcept(_S_construct(__a, __p, std::forward<_Args>(__args)...))) -> decltype(_S_construct(__a, __p, std::forward<_Args>(__args)...))
- static constexpr void `deallocate` (_Alloc &__a, pointer __p, size_type __n)
- static constexpr void `deallocate` (_Alloc &__a, pointer __p, size_type __n)
- template<typename `_Ptr` >
static constexpr std::enable_if_t< __is_custom_pointer< _Ptr >::value > **destroy** (_Alloc &__a, _Ptr __p) noexcept(noexcept(_Base_type::destroy(__a, std::to_address(__p))))
- template<typename `_Tp` >
static constexpr void **destroy** (_Alloc &__a, _Tp *__p) noexcept(noexcept(_S_destroy(__a, __p, 0)))
- static constexpr size_type `max_size` (const _Alloc &__a) noexcept
- static constexpr _Alloc `select_on_container_copy_construction` (const _Alloc &__rhs)

6.5.1 Detailed Description

template<typename `_Alloc`, typename = typename `_Alloc::value_type`>
struct `__gnu_cxx::__alloc_traits<_Alloc, typename>`

Uniform interface to C++98 and C++11 allocators.

6.5.2 Member Typedef Documentation

const_void_pointer

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::const_void_pointer = typename _Ptr<__cv_pointer, const
void>::type [inherited]
```

The allocator's const void pointer type.

`Alloc::const_void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<const void>`

is_always_equal

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::is_always_equal = __detected_or_t<typename is_empty<_↵
Alloc>::type, __equal, _Alloc> [inherited]
```

Whether all instances of the allocator type compare equal.

`Alloc::is_always_equal` if that type exists, otherwise `is_empty<Alloc>::type`

propagate_on_container_copy_assignment

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::propagate_on_container_copy_assignment = __detected_or_↵
t<false_type, __pocca, _Alloc> [inherited]
```

How the allocator is propagated on copy assignment.

`Alloc::propagate_on_container_copy_assignment` if that type exists, otherwise `false_type`

propagate_on_container_move_assignment

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::propagate_on_container_move_assignment = __detected_or_↵
t<false_type, __pocma, _Alloc> [inherited]
```

How the allocator is propagated on move assignment.

`Alloc::propagate_on_container_move_assignment` if that type exists, otherwise `false_type`

propagate_on_container_swap

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::propagate_on_container_swap = __detected_or_t<false_type,
__pocs, _Alloc> [inherited]
```

How the allocator is propagated on swap.

`Alloc::propagate_on_container_swap` if that type exists, otherwise `false_type`

void_pointer

```
template<typename _Alloc >
using std::allocator\_traits< _Alloc >::void_pointer = typename _Ptr<__v_pointer, void>::type
[inherited]
```

The allocator's void pointer type.

`Alloc::void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<void>`

6.5.3 Member Function Documentation

allocate() [1/4]

```
template<typename _Alloc >
static constexpr pointer std::allocator\_traits< _Alloc >::allocate (
```

```

    _Alloc & __a,
    size_type __n ) [inline], [static], [constexpr], [inherited]

```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.

Calls `a.allocate(n)`

`allocate()` [2/4]

```

template<typename _Alloc , typename = typename _Alloc::value_type>
static constexpr pointer std::allocator_traits<_Alloc >::allocate (
    _Alloc & __a,
    size_type __n ) [inline], [static], [constexpr]

```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.

Calls `a.allocate(n)`

`allocate()` [3/4]

```

template<typename _Alloc >
static constexpr pointer std::allocator_traits<_Alloc >::allocate (
    _Alloc & __a,
    size_type __n,
    const_void_pointer __hint ) [inline], [static], [constexpr], [inherited]

```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for *n* objects of type `value_type`

Returns `a.allocate(n, hint)` if that expression is well-formed, otherwise returns `a.allocate(n)`

`allocate()` [4/4]

```

template<typename _Alloc , typename = typename _Alloc::value_type>

```

```
static constexpr pointer std::allocator_traits< _Alloc >::allocate (
    _Alloc & __a,
    size_type __n,
    const_void_pointer __hint ) [inline], [static], [constexpr]
```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for *n* objects of type `value_type`

Returns `a.allocate(n, hint)` if that expression is well-formed, otherwise returns `a.allocate(n)`

construct()

```
template<typename _Alloc , typename = typename _Alloc::value_type>
template<typename _Tp , typename... _Args>
static constexpr auto std::allocator_traits< _Alloc >::construct (
    _Alloc & __a,
    _Tp * __p,
    _Args &&... __args ) -> decltype(_S_construct(__a, __p, std::forward<_Args>(__args)...) ) [inline], [static], [constexpr], [noexcept]
```

Construct an object of type `_Tp`

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for <code>Tp</code>
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<Args>(__args)...) if that expression is well-formed, otherwise uses placement-new to construct an object of type _Tp at location __p from the arguments __args...`

deallocate() [1/2]

```
template<typename _Alloc >
static constexpr void std::allocator_traits< _Alloc >::deallocate (
    _Alloc & __a,
    pointer __p,
    size_type __n ) [inline], [static], [constexpr], [inherited]
```

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.

Parameters

<code>__n</code>	The number of objects space was allocated for.
------------------	--

Calls `a.deallocate(p, n)`

deallocate() [2/2]

```
template<typename _Alloc , typename = typename _Alloc::value_type>
static constexpr void std::allocator_traits< _Alloc >::deallocate (
    _Alloc & __a,
    pointer __p,
    size_type __n ) [inline], [static], [constexpr]
```

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

destroy()

```
template<typename _Alloc , typename = typename _Alloc::value_type>
template<typename _Tp >
static constexpr void std::allocator_traits< _Alloc >::destroy (
    _Alloc & __a,
    _Tp * __p ) [inline], [static], [constexpr], [noexcept]
```

Destroy an object of type `_Tp`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Calls `__a.destroy(__p)` if that expression is well-formed, otherwise calls `__p->~_Tp()`

max_size()

```
template<typename _Alloc , typename = typename _Alloc::value_type>
static constexpr size_type std::allocator_traits< _Alloc >::max_size (
    const _Alloc & __a ) [inline], [static], [constexpr], [noexcept]
```

The maximum supported allocation size.

Parameters

<code>__a</code>	An allocator.
------------------	---------------

Returns

`__a.max_size()` or `numeric_limits<size_type>::max()`

Returns `__a.max_size()` if that expression is well-formed, otherwise returns `numeric_limits<size_type>::max()`

Referenced by `std::forward_list<_Tp, _Alloc>::max_size()`, and `std::list<_Tp, _Alloc>::max_size()`.

select_on_container_copy_construction()

```
template<typename _Alloc >
static constexpr _Alloc std::allocator_traits<_Alloc>::select_on_container_copy_construction (
    const _Alloc & __rhs ) [inline], [static], [constexpr], [inherited]
```

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs.select_on_container_copy_construction()` or `__rhs`

Returns `__rhs.select_on_container_copy_construction()` if that expression is well-formed, otherwise returns `__rhs`

The documentation for this struct was generated from the following file:

- [ext/alloc_traits.h](#)

6.6 std::__atomic_base<_ITp> Struct Template Reference

```
#include <atomic_base.h>
```

Public Types

- using **difference_type** = `value_type`
- using **value_type** = `_ITp`

Public Member Functions

- `constexpr __atomic_base (__int_type __i) noexcept`
- `__atomic_base (const __atomic_base &)=delete`
- `bool compare_exchange_strong (__int_type &__i1, __int_type __i2, memory_order __m1, memory_order __m2) noexcept`
- `bool compare_exchange_strong (__int_type &__i1, __int_type __i2, memory_order __m1, memory_order __m2) volatile noexcept`
- `bool compare_exchange_strong (__int_type &__i1, __int_type __i2, memory_order __m=memory_order_seq_cst) noexcept`
- `bool compare_exchange_strong (__int_type &__i1, __int_type __i2, memory_order __m=memory_order_seq_cst) volatile noexcept`

- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, [memory_order](#) __m1, [memory_order](#) __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, [memory_order](#) __m1, [memory_order](#) __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** ([memory_order](#) __m=memory_order_seq_cst) const noexcept
- __int_type **load** ([memory_order](#) __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- __int_type **operator=** (__int_type __i) noexcept
- __int_type **operator=** (__int_type __i) volatile noexcept
- [__atomic_base](#) & **operator=** (const [__atomic_base](#) &) volatile=delete
- [__atomic_base](#) & **operator=** (const [__atomic_base](#) &)=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept
- void **store** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- void **store** (__int_type __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- void **wait** (__int_type __old, [memory_order](#) __m=memory_order_seq_cst) const noexcept

6.6.1 Detailed Description

```
template<typename _ITp>
struct std::__atomic_base< _ITp >
```

Base class for atomic integrals.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

6.7 std::__atomic_base< _PTp * > Struct Template Reference

```
#include <atomic_base.h>
```

Public Member Functions

- constexpr **__atomic_base** (__pointer_type __p) noexcept
- **__atomic_base** (const **__atomic_base** &)=delete
- bool **compare_exchange_strong** (__pointer_type &__p1, __pointer_type __p2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__pointer_type &__p1, __pointer_type __p2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__pointer_type &__p1, __pointer_type __p2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__pointer_type &__p1, __pointer_type __p2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- __pointer_type **exchange** (__pointer_type __p, **memory_order** __m=memory_order_seq_cst) noexcept
- __pointer_type **exchange** (__pointer_type __p, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __pointer_type **fetch_add** (ptrdiff_t __d, **memory_order** __m=memory_order_seq_cst) noexcept
- __pointer_type **fetch_add** (ptrdiff_t __d, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __pointer_type **fetch_sub** (ptrdiff_t __d, **memory_order** __m=memory_order_seq_cst) noexcept
- __pointer_type **fetch_sub** (ptrdiff_t __d, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __pointer_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __pointer_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () const noexcept
- void **notify_one** () const noexcept
- **operator __pointer_type** () const noexcept
- **operator __pointer_type** () const volatile noexcept
- __pointer_type **operator++** () noexcept
- __pointer_type **operator++** () volatile noexcept
- __pointer_type **operator++** (int) noexcept
- __pointer_type **operator++** (int) volatile noexcept
- __pointer_type **operator+=** (ptrdiff_t __d) noexcept
- __pointer_type **operator+=** (ptrdiff_t __d) volatile noexcept
- __pointer_type **operator--** () noexcept
- __pointer_type **operator--** () volatile noexcept
- __pointer_type **operator--** (int) noexcept
- __pointer_type **operator--** (int) volatile noexcept
- __pointer_type **operator-=** (ptrdiff_t __d) noexcept
- __pointer_type **operator-=** (ptrdiff_t __d) volatile noexcept
- __pointer_type **operator=** (__pointer_type __p) noexcept

- `__pointer_type operator= (__pointer_type __p) volatile noexcept`
- `__atomic_base & operator= (const __atomic_base &) volatile=delete`
- `__atomic_base & operator= (const __atomic_base &)=delete`
- `void store (__pointer_type __p, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__pointer_type __p, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__pointer_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

6.7.1 Detailed Description

```
template<typename _PTp>
struct std::__atomic_base<_PTp * >
```

Partial specialization for pointer types.

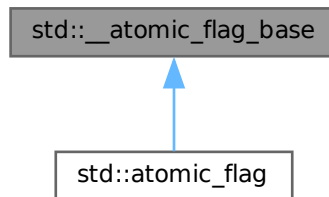
The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

6.8 std::__atomic_flag_base Struct Reference

```
#include <atomic_base.h>
```

Inheritance diagram for std::__atomic_flag_base:



Public Attributes

- `__atomic_flag_data_type _M_i`

6.8.1 Detailed Description

Base type for atomic_flag.

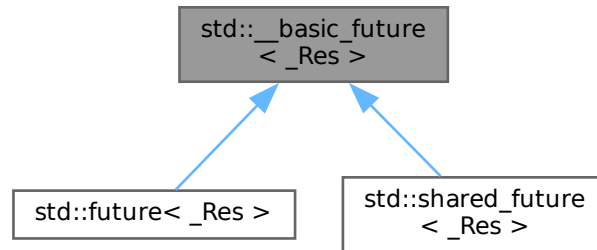
Base type is POD with data, allowing atomic_flag to derive from it and meet the standard layout type requirement. In addition to compatibility with a C interface, this allows different implementations of atomic_flag to use the same atomic operation functions, via a standard conversion to the __atomic_flag_base argument.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

6.9 std::__basic_future< _Res > Class Template Reference

Inheritance diagram for std::__basic_future< _Res >:



Public Member Functions

- **__basic_future** (const **__basic_future** &)=delete
- **__basic_future** & **operator=** (const **__basic_future** &)=delete
- bool **valid** () const noexcept
- void **wait** () const
- template<typename _Rep, typename _Period >
future_status wait_for (const chrono::duration< _Rep, _Period > &__rel) const
- template<typename _Clock, typename _Duration >
future_status wait_until (const chrono::time_point< _Clock, _Duration > &__abs) const

Protected Types

- typedef __future_base::Result< _Res > & **__result_type**
- typedef **shared_ptr**< _State_base > **__state_type**

Protected Member Functions

- **__basic_future** (const **__state_type** &__state)
- **__basic_future** (const **shared_future**< _Res > &) noexcept
- **__basic_future** (**future**< _Res > &&) noexcept
- **__basic_future** (**shared_future**< _Res > &&) noexcept
- **__result_type** **_M_get_result** () const
- void **_M_swap** (**__basic_future** &__that) noexcept

6.9.1 Detailed Description

```

template<typename _Res>
class std::__basic_future< _Res >

```

Common implementation for future and shared_future.

6.9.2 Member Function Documentation

`_M_get_result()`

```
template<typename _Res >
__result_type std::__basic_future<_Res>::_M_get_result ( ) const [inline], [protected]
```

Wait for the state to be ready and rethrow any stored exception.

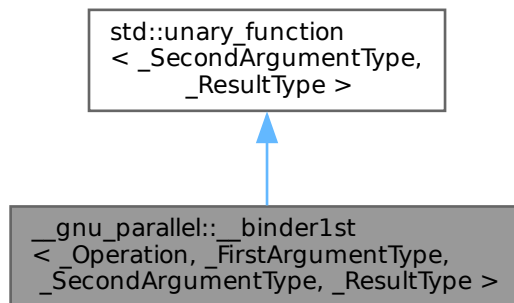
The documentation for this class was generated from the following file:

- [future](#)

6.10 `__gnu_parallel::__binder1st<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType>` Class Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::__binder1st<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType>`:



Public Types

- typedef `_SecondArgumentType` [argument_type](#)
- typedef `_ResultType` [result_type](#)

Public Member Functions

- `_binder1st` (`const _Operation &__x`, `const _FirstArgumentType &__y`)
- `_ResultType operator()` (`_SecondArgumentType &__x`) `const`
- `_ResultType operator()` (`const _SecondArgumentType &__x`)

Protected Attributes

- `_Operation` `_M_op`
- `_FirstArgumentType` `_M_value`

6.10.1 Detailed Description

```
template<typename _Operation, typename _FirstArgumentType, typename _SecondArgumentType, typename
_ResultType>
```

```
class __gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >
```

Similar to `std::binder1st`, but giving the argument types explicitly.

6.10.2 Member Typedef Documentation

argument_type

```
typedef _SecondArgumentType std::unary_function< _SecondArgumentType , _ResultType >::argument_↵
type [inherited]
```

`argument_type` is the type of the argument

result_type

```
typedef _ResultType std::unary_function< _SecondArgumentType , _ResultType >::result_type [inherited]
```

`result_type` is the return type

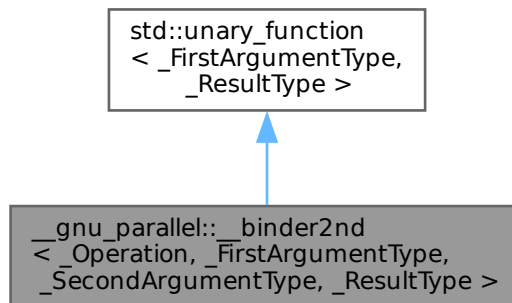
The documentation for this class was generated from the following file:

- [base.h](#)

6.11 __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType > Class Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`:



Public Types

- typedef `_FirstArgumentType` [argument_type](#)
- typedef `_ResultType` [result_type](#)

Public Member Functions

- **__binder2nd** (const _Operation &__x, const _SecondArgumentType &__y)
- _ResultType **operator()** (_FirstArgumentType &__x)
- _ResultType **operator()** (const _FirstArgumentType &__x) const

Protected Attributes

- _Operation **_M_op**
- _SecondArgumentType **_M_value**

6.11.1 Detailed Description

template<typename _Operation, typename _FirstArgumentType, typename _SecondArgumentType, typename _ResultType>
class __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >

Similar to std::binder2nd, but giving the argument types explicitly.

6.11.2 Member Typedef Documentation

argument_type

```
typedef _FirstArgumentType std::unary_function< _FirstArgumentType , _ResultType >::argument_type
[inherited]
```

argument_type is the type of the argument

result_type

```
typedef _ResultType std::unary_function< _FirstArgumentType , _ResultType >::result_type [inherited]
result_type is the return type
```

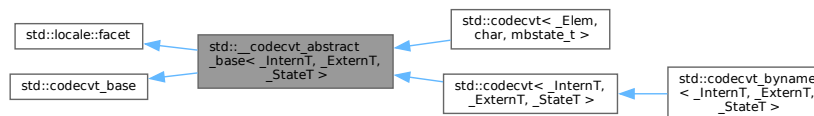
The documentation for this class was generated from the following file:

- [base.h](#)

6.12 std::__codecvt_abstract_base< _InternT, _ExternT, _StateT > Class Template Reference

```
#include <codecvt.h>
```

Inheritance diagram for std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >:



Public Types

- typedef _ExternT **extern_type**
- typedef _InternT **intern_type**
- typedef codecvt_base::result **result**
- typedef _StateT **state_type**

Public Member Functions

- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Protected Member Functions

- **__codecvt_abstract_base** (size_t __refs=0)
- virtual bool **do_always_noconv** () const throw () =0
- virtual int **do_encoding** () const throw () =0
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const =0
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const =0
- virtual int **do_max_length** () const throw () =0
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const =0
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const =0

Static Protected Member Functions

- static **__c_locale _S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static **__c_locale _S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static **__c_locale _S_lc_type_c_locale** (__c_locale __cloc, const char *__s)

6.12.1 Detailed Description

```
template<typename _InternT, typename _ExternT, typename _StateT>
class std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >
```

Common base for codecv functions.

This template class provides implementations of the public functions that forward to the protected virtual functions.

This template also provides abstract stubs for the protected virtual functions.

6.12.2 Member Function Documentation

do_out()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type * __from_next,
```

```
extern_type * __to,
extern_type * __to_end,
extern_type *& __to_next ) const [protected], [pure virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implemented in `std::codecvt< _InternT, _ExternT, _StateT >`, `std::codecvt< char, char, mbstate_t >`, `std::codecvt< wchar_t, char, mbstate_t >`, `std::codecvt< char16_t, char, mbstate_t >`, `std::codecvt< char32_t, char, mbstate_t >`, and `std::codecvt< _InternT, _ExternT, encoding_t >`.
Referenced by `std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::out()`.

in()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline]
```

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

out()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type * __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

References [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::do_out\(\)](#).

unshift()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.13 `__gnu_cxx::__common_pool_policy<_PoolTp, _Thread >` Struct Template Reference

```
#include <mt_allocator.h>
```

Inherits `__gnu_cxx::__common_pool_base<_PoolTp, _Thread >`.

6.13.1 Detailed Description

```
template<template< bool > class _PoolTp, bool _Thread>
struct __gnu_cxx::__common_pool_policy<_PoolTp, _Thread >
```

Policy for shared `__pool` objects.

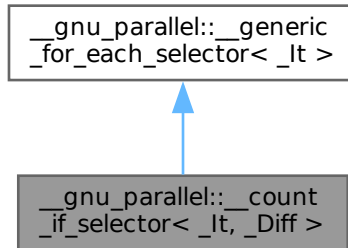
The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

6.14 `__gnu_parallel::__count_if_selector<_It, _Diff >` Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__count_if_selector<_It, _Diff>`:



Public Member Functions

- `template<typename _Op>`
`_Diff operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.14.1 Detailed Description

```
template<typename _It, typename _Diff>
struct __gnu_parallel::__count_if_selector<_It, _Diff>
```

`std::count_if()` selector.

6.14.2 Member Function Documentation

`operator()()`

```
template<typename _It, typename _Diff>
template<typename _Op>
_Diff __gnu_parallel::__count_if_selector<_It, _Diff>::operator() (
    _Op & __o,
    _It __i ) [inline]
```

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

Returns

1 if count, 0 if does not count.

6.14.3 Member Data Documentation**_M_finish_iterator**

```
template<typename _It >
```

```
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
```

_Iterator on last element processed; needed for some algorithms (e. g. std::transform()).

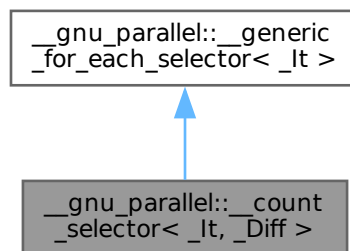
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.15 __gnu_parallel::__count_selector<_It, _Diff> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for __gnu_parallel::__count_selector<_It, _Diff>:

**Public Member Functions**

- `template<typename _ValueType >`
`_Diff operator\(\) (_ValueType &__v, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.15.1 Detailed Description

```
template<typename _It, typename _Diff>
struct __gnu_parallel::__count_selector<_It, _Diff>
```

std::count() selector.

6.15.2 Member Function Documentation**operator>()()**

```
template<typename _It , typename _Diff >
template<typename _ValueType >
```

```

_Diff __gnu_parallel::__count_selector< _It, _Diff >::operator() (
    _ValueType & __v,
    _It __i ) [inline]

```

Functor execution.

Parameters

<code>__v</code>	Current value.
<code>__i</code>	iterator referencing object.

Returns

1 if count, 0 if does not count.

6.15.3 Member Data Documentation

`_M_finish_iterator`

```

template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).

```

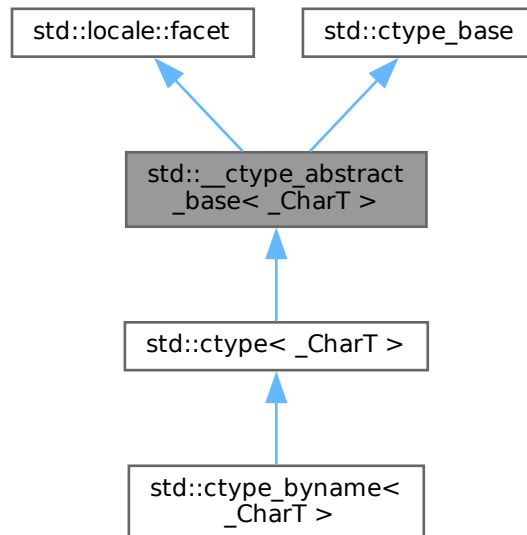
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.16 `std::__ctype_abstract_base< _CharT >` Class Template Reference

```
#include <locale_facets.h>
```

Inheritance diagram for `std::__ctype_abstract_base< _CharT >`:



Public Types

- typedef const int * **__to_type**
- typedef _CharT **char_type**
- typedef unsigned short **mask**

Public Member Functions

- const **char_type** * **is** (const **char_type** * __lo, const **char_type** * __hi, mask * __vec) const
- bool **is** (mask __m, **char_type** __c) const
- char **narrow** (**char_type** __c, char __dfault) const
- const **char_type** * **narrow** (const **char_type** * __lo, const **char_type** * __hi, char __dfault, char * __to) const
- const **char_type** * **scan_is** (mask __m, const **char_type** * __lo, const **char_type** * __hi) const
- const **char_type** * **scan_not** (mask __m, const **char_type** * __lo, const **char_type** * __hi) const
- const **char_type** * **tolower** (**char_type** * __lo, const **char_type** * __hi) const
- **char_type** **tolower** (**char_type** __c) const
- const **char_type** * **toupper** (**char_type** * __lo, const **char_type** * __hi) const
- **char_type** **toupper** (**char_type** __c) const
- **char_type** **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, **char_type** * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- **__ctype_abstract_base** (size_t __refs=0)
- virtual const **char_type** * **do_is** (const **char_type** * __lo, const **char_type** * __hi, mask * __vec) const =0
- virtual bool **do_is** (mask __m, **char_type** __c) const =0
- virtual char **do_narrow** (**char_type** __c, char __dfault) const =0
- virtual const **char_type** * **do_narrow** (const **char_type** * __lo, const **char_type** * __hi, char __dfault, char * __to) const =0
- virtual const **char_type** * **do_scan_is** (mask __m, const **char_type** * __lo, const **char_type** * __hi) const =0
- virtual const **char_type** * **do_scan_not** (mask __m, const **char_type** * __lo, const **char_type** * __hi) const =0
- virtual const **char_type** * **do_tolower** (**char_type** * __lo, const **char_type** * __hi) const =0
- virtual **char_type** **do_tolower** (**char_type** __c) const =0
- virtual const **char_type** * **do_toupper** (**char_type** * __lo, const **char_type** * __hi) const =0
- virtual **char_type** **do_toupper** (**char_type** __c) const =0
- virtual **char_type** **do_widen** (char __c) const =0
- virtual const char * **do_widen** (const char * __lo, const char * __hi, **char_type** * __to) const =0

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_ctype_c_locale (__c_locale __cloc, const char *__s)`

6.16.1 Detailed Description

```
template<typename _CharT>
class std::__ctype_abstract_base< _CharT >
```

Common base for ctype facet.

This template class provides implementations of the public functions that forward to the protected virtual functions.

This template also provides abstract stubs for the protected virtual functions.

6.16.2 Member Typedef Documentation

char_type

```
template<typename _CharT >
typedef _CharT std::__ctype_abstract_base< _CharT >::char_type
```

Typedef for the template parameter.

6.16.3 Member Function Documentation

do_is() [1/2]

```
template<typename _CharT >
virtual const char_type * std::__ctype_abstract_base< _CharT >::do_is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [protected], [pure virtual]
```

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the input.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implemented in `std::ctype< _CharT >`, and `std::ctype< wchar_t >`.

do_is() [2/2]

```
template<typename _CharT >
```

```
virtual bool std::__ctype_abstract_base<_CharT>::do_is (
    mask __m,
    char_type __c ) const [protected], [pure virtual]
```

Test char_type classification.

This function finds a mask *M* for *c* and compares it to mask *m*.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

\leftrightarrow __c	The char_type to find the mask of.
\leftrightarrow __m	The mask to compare against.

Returns

(*M* & __m) != 0.

Implemented in [std::ctype<_CharT>](#), and [std::ctype<wchar_t>](#).

Referenced by [std::__ctype_abstract_base<_CharT>::is\(\)](#), and [std::__ctype_abstract_base<_CharT>::is\(\)](#).

do_narrow() [1/2]

```
template<typename _CharT>
virtual char std::__ctype_abstract_base<_CharT>::do_narrow (
    char_type __c,
    char __default ) const [protected], [pure virtual]
```

Narrow char_type to char.

This virtual function converts the argument to char using the simplest reasonable transformation. If the conversion fails, default is returned instead.

do_narrow() is a hook for a derived facet to change the behavior of narrowing. do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

__c	The char_type to convert.
__default	Char to return if conversion fails.

Returns

The converted char.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

Referenced by [std::__ctype_abstract_base<_CharT>::narrow\(\)](#), and [std::__ctype_abstract_base<_CharT>::narrow\(\)](#).

do_narrow() [2/2]

```
template<typename _CharT>
virtual const char_type * std::__ctype_abstract_base<_CharT>::do_narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [protected], [pure virtual]
```


Narrow `char_type` array to `char`.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__dfault` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implemented in `std::ctype< _CharT >`, and `std::ctype< wchar_t >`.

`do_scan_is()`

```
template<typename _CharT >
virtual const char_type * std::__ctype_abstract_base< _CharT >::do_scan_is (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [protected], [pure virtual]
```

Find `char_type` matching mask.

This function searches for and returns the first `char_type` `c` in `[__lo,__hi)` for which `is(__m,c)` is true.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching `char_type` if found, else `__hi`.

Implemented in `std::ctype< _CharT >`, and `std::ctype< wchar_t >`.

Referenced by `std::__ctype_abstract_base< _CharT >::scan_is()`.

`do_scan_not()`

```
template<typename _CharT >
virtual const char_type * std::__ctype_abstract_base< _CharT >::do_scan_not (
    mask __m,
```

```
const char_type * __lo,
const char_type * __hi ) const [protected], [pure virtual]
```

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>↔ _m</code>	The mask to compare against.
<code>↔ _lo</code>	Pointer to start of range.
<code>↔ _hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char_type if found, else __hi.

Implemented in `std::ctype<_CharT>`, and `std::ctype<wchar_t>`.

Referenced by `std::__ctype_abstract_base<_CharT>::scan_not()`.

do_tolower() [1/2]

```
template<typename _CharT>
virtual const char_type * std::__ctype_abstract_base<_CharT>::do_tolower (
    char_type * __lo,
    const char_type * __hi ) const [protected], [pure virtual]
```

Convert array to lowercase.

This virtual function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

<code>↔ _lo</code>	Pointer to start of range.
<code>↔ _hi</code>	Pointer to end of range.

Returns

__hi.

Implemented in `std::ctype<_CharT>`, and `std::ctype<wchar_t>`.

do_tolower() [2/2]

```
template<typename _CharT>
virtual char_type std::__ctype_abstract_base<_CharT>::do_tolower (
    char_type __c ) const [protected], [pure virtual]
```

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The lowercase `char_type` if convertible, else `__c`.

Implemented in [std::ctype< _CharT >](#), and [std::ctype< wchar_t >](#).

Referenced by [std::__ctype_abstract_base< _CharT >::tolower\(\)](#), and [std::__ctype_abstract_base< _CharT >::tolower\(\)](#).

`do_toupper()` [1/2]

```
template<typename _CharT >
virtual const char_type * std::__ctype_abstract_base< _CharT >::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const [protected], [pure virtual]
```

Convert array to uppercase.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implemented in [std::ctype< _CharT >](#), and [std::ctype< wchar_t >](#).

`do_toupper()` [2/2]

```
template<typename _CharT >
virtual char_type std::__ctype_abstract_base< _CharT >::do_toupper (
    char_type __c ) const [protected], [pure virtual]
```

Convert to uppercase.

This virtual function converts the `char_type` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

 <code>__c</code>	The char_type to convert.
---	---------------------------

Returns

The uppercase char_type if convertible, else `__c`.

Implemented in [std::ctype<_CharT>](#), and [std::ctype<wchar_t>](#).

Referenced by [std::__ctype_abstract_base<_CharT>::toupper\(\)](#), and [std::__ctype_abstract_base<_CharT>::toupper\(\)](#).

do_widen() [1/2]

```
template<typename _CharT >
virtual char_type std::__ctype_abstract_base<_CharT >::do_widen (
    char __c ) const [protected], [pure virtual]
```


Widen char.

This virtual function converts the char to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__c</code>	The char to convert.
---	----------------------

Returns

The converted char_type

Implemented in [std::ctype<_CharT>](#), and [std::ctype<wchar_t>](#).

Referenced by [std::__ctype_abstract_base<_CharT>::widen\(\)](#), and [std::__ctype_abstract_base<_CharT>::widen\(\)](#).

do_widen() [2/2]

```
template<typename _CharT >
virtual const char * std::__ctype_abstract_base<_CharT >::do_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [protected], [pure virtual]
```




Widen char array.

This function converts each char in the input to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__lo</code>	Pointer to start range.
 <code>__hi</code>	Pointer to end of range.
 <code>__to</code>	Pointer to the destination array.

Returns`__hi`.

Implemented in [std::ctype<_CharT>](#), and [std::ctype<wchar_t>](#).

is() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT >::is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [inline]
```

Return a mask array.

This function finds the mask for each `char_type` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the char array. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_is\(\)](#).

is() [2/2]

```
template<typename _CharT >
bool std::__ctype_abstract_base<_CharT >::is (
    mask __m,
    char_type __c ) const [inline]
```

Test `char_type` classification.

This function finds a mask `M` for `__c` and compares it to mask `__m`. It does so by returning the value of `ctype<char_↵type>::do_is()`.

Parameters

<code>_↵c</code>	The <code>char_type</code> to compare the mask of.
<code>_↵m</code>	The mask to compare against.

Returns`(M & __m) != 0`.

References [std::__ctype_abstract_base<_CharT>::do_is\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::time_get<_CharT, _InIter>::get\(\)](#).

narrow() [1/2]

```
template<typename _CharT >
```

```
char std::__ctype_abstract_base<_CharT>::narrow (
    char_type __c,
    char __default ) const [inline]
```

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, default is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), and [std::time_put<_CharT, _OutIter>::put\(\)](#).

narrow() [2/2]

```
template<typename _CharT>
const char_type * std::__ctype_abstract_base<_CharT>::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline]
```

Narrow array to char array.

This function converts each char_type in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char_type in the input that cannot be converted, default is used instead. It does so by returning ctype<char_type>::do_narrow(__lo, __hi, __default, __to).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

scan_is()




```
template<typename _CharT>
const char_type * std::__ctype_abstract_base<_CharT>::scan_is (
    mask __m,
```

```
const char_type * __lo,
const char_type * __hi ) const [inline]
```

Find char_type matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is true. It does so by returning ctype<char_type>::do_scan_is().

Parameters

 _m	The mask to compare against.
 _lo	Pointer to start of range.
 _hi	Pointer to end of range.

Returns

Pointer to matching char_type if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_is\(\)](#).




scan_not()

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline]
```

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

 _m	The mask to compare against.
 _lo	Pointer to first char in range.
 _hi	Pointer to end of range.

Returns

Pointer to non-matching char if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_not\(\)](#).

tolower() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline]
```

Convert array to lowercase.

This function converts each `char_type` in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_tolower(__lo, __hi)`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_tolower\(\)](#).

tolower() [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT>::tolower (
    char_type __c ) const [inline]
```

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_tolower(c)`.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The lowercase `char_type` if convertible, else `__c`.

References [std::__ctype_abstract_base<_CharT>::do_tolower\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get\(\)](#).

toupper() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline]
```

Convert array to uppercase.

This function converts each `char_type` in the range `[lo,hi)` to uppercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_toupper(lo, hi)`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns`__hi`.References [std::__ctype_abstract_base<_CharT>::do_toupper\(\)](#).**toupper()** [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT>::toupper (
    char_type __c ) const [inline]
```

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_toupper()`.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

ReturnsThe uppercase `char_type` if convertible, else `__c`.References [std::__ctype_abstract_base<_CharT>::do_toupper\(\)](#).Referenced by [std::time_get<_CharT, _Inlter>::get\(\)](#).**widen()** [1/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT>::widen (
    char __c ) const [inline]
```

Widen char to `char_type`.

This function converts the char argument to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

ReturnsThe converted `char_type`.References [std::__ctype_abstract_base<_CharT>::do_widen\(\)](#).

Referenced by [std::money_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get\(\)](#), [std::money_put<_CharT, _Outlter>::do_put\(\)](#) and [std::time_put<_CharT, _Outlter>::do_put\(\)](#).

widen() [2/2]

```
template<typename _CharT >
const char * std::__ctype_abstract_base<_CharT>::widen (
    const char * __lo,
```




```
const char * __hi,
char_type * __to ) const [inline]
```

Widen array to char_type.

This function converts each char in the input to char_type using the simplest reasonable transformation. It does so by returning ctype<char_type>::do_widen(c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

 __lo	Pointer to start of range.
 __hi	Pointer to end of range.
 __to	Pointer to the destination array.

Returns

[!\[\]\(6059a5aa8b4ca7bb793408023d6c6e42_img.jpg\) __hi](#).

References [std::__ctype_abstract_base<_CharT>::do_widen\(\)](#).

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.17 std::filesystem::__directory_iterator_proxy Struct Reference

```
#include <fs_dir.h>
```

Public Member Functions

- [directory_entry operator*](#) () &&noexcept
- const [directory_entry & operator*](#) () const &&noexcept

Friends

- class [directory_iterator](#)
- class [recursive_directory_iterator](#)

6.17.1 Detailed Description

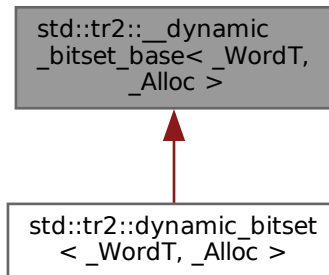
Proxy returned by post-increment on directory iterators.

The documentation for this struct was generated from the following file:

- [bits/fs_dir.h](#)

6.18 std::tr2::__dynamic_bitset_base< _WordT, _Alloc > Struct Template Reference

Inheritance diagram for std::tr2::__dynamic_bitset_base< _WordT, _Alloc >:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_WordT` **block_type**
- typedef `size_t` **size_type**

Public Member Functions

- `__dynamic_bitset_base` (`__dynamic_bitset_base` && __b)=default
- `__dynamic_bitset_base` (const `__dynamic_bitset_base` &)=default
- `__dynamic_bitset_base` (const `allocator_type` &__alloc)
- `__dynamic_bitset_base` (`size_type` __nbits, unsigned long long __val=0ULL, const `allocator_type` &__alloc=allocator_type())
- `size_t` **M_are_all_aux** () const noexcept
- void **M_clear** () noexcept
- void **M_do_and** (const `__dynamic_bitset_base` &__x) noexcept
- void **M_do_append_block** (`block_type` __block, `size_type` __pos)
- `size_t` **M_do_count** () const noexcept
- void **M_do_dif** (const `__dynamic_bitset_base` &__x) noexcept
- `size_type` **M_do_find_first** (`size_t` __not_found) const
- `size_type` **M_do_find_next** (`size_t` __prev, `size_t` __not_found) const
- void **M_do_flip** () noexcept
- void **M_do_left_shift** (`size_t` __shift)
- void **M_do_or** (const `__dynamic_bitset_base` &__x) noexcept
- void **M_do_reset** () noexcept
- void **M_do_right_shift** (`size_t` __shift)
- void **M_do_set** () noexcept
- unsigned long long **M_do_to_ullong** () const
- unsigned long **M_do_to_ulong** () const
- void **M_do_xor** (const `__dynamic_bitset_base` &__x) noexcept
- `allocator_type` **M_get_allocator** () const noexcept
- `block_type` **M_getword** (`size_type` __pos) const noexcept

- block_type & **_M_getword** (size_type __pos) noexcept
- block_type **_M_hiword** () const noexcept
- block_type & **_M_hiword** () noexcept
- bool **_M_is_any** () const noexcept
- bool **_M_is_equal** (const __dynamic_bitset_base &__x) const noexcept
- bool **_M_is_less** (const __dynamic_bitset_base &__x) const noexcept
- bool **_M_is_proper_subset_of** (const __dynamic_bitset_base &__b) const noexcept
- bool **_M_is_subset_of** (const __dynamic_bitset_base &__b) noexcept
- void **_M_resize** (size_t __nbits, bool __value)
- size_type **_M_size** () const noexcept
- void **_M_swap** (__dynamic_bitset_base &__b) noexcept
- __dynamic_bitset_base & **operator=** (__dynamic_bitset_base &&)=default
- __dynamic_bitset_base & **operator=** (const __dynamic_bitset_base &)=default

Static Public Member Functions

- static block_type **_S_maskbit** (size_type __pos) noexcept
- static size_type **_S_whichbit** (size_type __pos) noexcept
- static size_type **_S_whichbyte** (size_type __pos) noexcept
- static size_type **_S_whichword** (size_type __pos) noexcept

Public Attributes

- [std::vector](#)< block_type, allocator_type > **_M_w**

Static Public Attributes

- static const size_type **_S_bits_per_block**
- static const size_type **npos**

6.18.1 Detailed Description

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
struct std::tr2::__dynamic_bitset_base< _WordT, _Alloc >
```

Base class, general case.

See documentation for dynamic_bitset.

6.18.2 Member Data Documentation

_M_w

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::vector<block_type, allocator_type> std::tr2::\_\_dynamic\_bitset\_base< _WordT, _Alloc >::_M_w
```

0 is the least significant word.

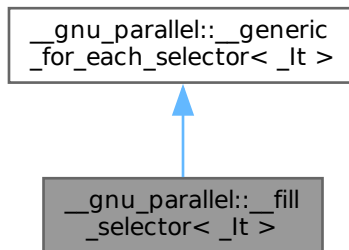
The documentation for this struct was generated from the following files:

- [dynamic_bitset](#)
- [dynamic_bitset.tcc](#)

6.19 __gnu_parallel::__fill_selector<_It> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for __gnu_parallel::__fill_selector<_It>:



Public Member Functions

- template<typename _ValueType >
bool [operator\(\)](#) (_ValueType &__v, _It __i)

Public Attributes

- _It [_M_finish_iterator](#)

6.19.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__fill_selector<_It>
```

std::fill() selector.

6.19.2 Member Function Documentation

operator>()

```
template<typename _It >
template<typename _ValueType >
bool __gnu_parallel::__fill_selector<_It>::operator() (
    _ValueType & __v,
    _It __i ) [inline]
```

Functor execution.

Parameters

__v	Current value.
__i	iterator referencing object.

6.19.3 Member Data Documentation

M_finish_iterator

```
template<typename _It >
```

```
_It __gnu_parallel::__generic_for_each_selector< _It >::__M_finish_iterator [inherited]  
_Iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

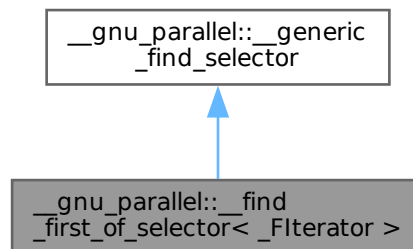
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.20 __gnu_parallel::__find_first_of_selector<_FIterator> Struct Template Reference

```
#include <find_selectors.h>
```

Inheritance diagram for __gnu_parallel::__find_first_of_selector<_FIterator>:



Public Member Functions

- **__find_first_of_selector** (_FIterator __begin, _FIterator __end)
- template<typename _RAIter1, typename _RAIter2, typename _Pred >
std::pair< _RAIter1, _RAIter2 > [_M_sequential_algorithm](#) (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)
- template<typename _RAIter1, typename _RAIter2, typename _Pred >
bool [operator\(\)](#) (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)

Public Attributes

- _FIterator **_M_begin**
- _FIterator **_M_end**

6.20.1 Detailed Description

```
template<typename _FIterator>
```

```
struct __gnu_parallel::__find_first_of_selector<_FIterator>
```

Test predicate on several elements.

6.20.2 Member Function Documentation

`_M_sequential_algorithm()`

```
template<typename _FIterator >
template<typename _RAIter1 , typename _RAIter2 , typename _Pred >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_first_of_selector< _FIterator >::_M↵
sequential_algorithm (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred ) [inline]
```

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

`operator>()()`

```
template<typename _FIterator >
template<typename _RAIter1 , typename _RAIter2 , typename _Pred >
bool __gnu_parallel::__find_first_of_selector< _FIterator >::operator() (
    _RAIter1 __i1,
    _RAIter2 __i2,
    _Pred __pred ) [inline]
```

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

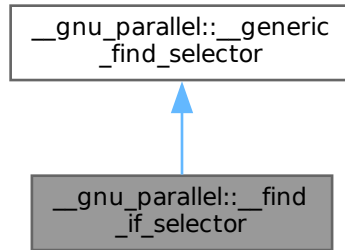
The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

6.21 `__gnu_parallel::__find_if_selector` Struct Reference

```
#include <find_selectors.h>
```

Inheritance diagram for __gnu_parallel::__find_if_selector:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair< _RAIter1, _RAIter2 > _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator\(\) (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

6.21.1 Detailed Description

Test predicate on a single element, used for `std::find()` and `std::find_if()`.

6.21.2 Member Function Documentation

[_M_sequential_algorithm\(\)](#)

```

template<typename _RAIter1, typename _RAIter2, typename _Pred >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_if_selector::_M_sequential_algorithm (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred ) [inline]
  
```

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

[operator\(\)\(\)](#)

```

template<typename _RAIter1, typename _RAIter2, typename _Pred >
bool __gnu_parallel::__find_if_selector::operator() (
  
```



```

    _RAIter1 __i1,
    _RAIter2 __i2,
    _Pred __pred ) [inline]

```

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

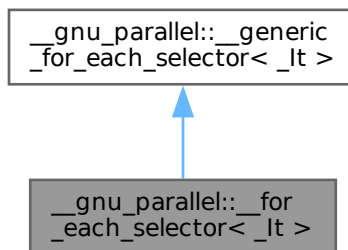
The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

6.22 __gnu_parallel::__for_each_selector<_It> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for __gnu_parallel::__for_each_selector<_It>:



Public Member Functions

- `template<typename _Op>`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.22.1 Detailed Description

```

template<typename _It>
struct __gnu_parallel::__for_each_selector<_It>

```

std::for_each() selector.



6.22.2 Member Function Documentation

operator>()

```
template<typename _It >
template<typename _Op >
bool __gnu_parallel::__for_each_selector< _It >::operator() (
    _Op & __o,
    _It __i ) [inline]
```

Functor execution.

Parameters

 <code>__o</code>	Operator.
 <code>__i</code>	iterator referencing object.

6.22.3 Member Data Documentation

_M_finish_iterator

```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.23 __cxxabiv1::__forced_unwind Class Reference

```
#include <cxxabi_forced.h>
```

6.23.1 Detailed Description

Thrown as part of forced unwinding.

A magic placeholder class that can be caught by reference to recognize forced unwinding.

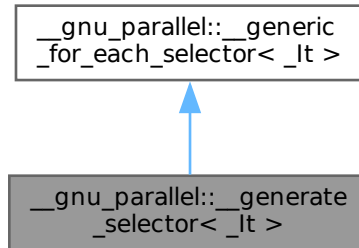
The documentation for this class was generated from the following file:

- [cxxabi_forced.h](#)

6.24 __gnu_parallel::__generate_selector< _It > Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__generate_selector<_It>`:



Public Member Functions

- `template<typename _Op >`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.24.1 Detailed Description

`template<typename _It>`
`struct __gnu_parallel::__generate_selector<_It>`

`std::generate()` selector.

6.24.2 Member Function Documentation

`operator()()`

```

template<typename _It >
template<typename _Op >
bool __gnu_parallel::__generate_selector<_It >::operator() (
    _Op & __o,
    _It __i ) [inline]
  
```

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

6.24.3 Member Data Documentation

M_finish_iterator

```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

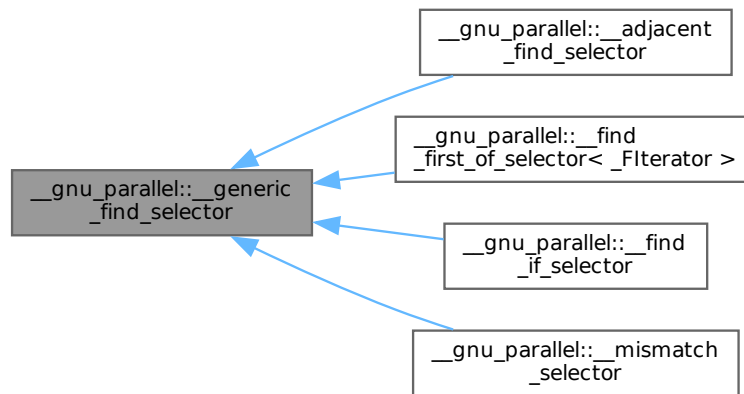
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.25 __gnu_parallel::__generic_find_selector Struct Reference

```
#include <find_selectors.h>
```

Inheritance diagram for __gnu_parallel::__generic_find_selector:



6.25.1 Detailed Description

Base class of all __gnu_parallel::__find_template selectors.

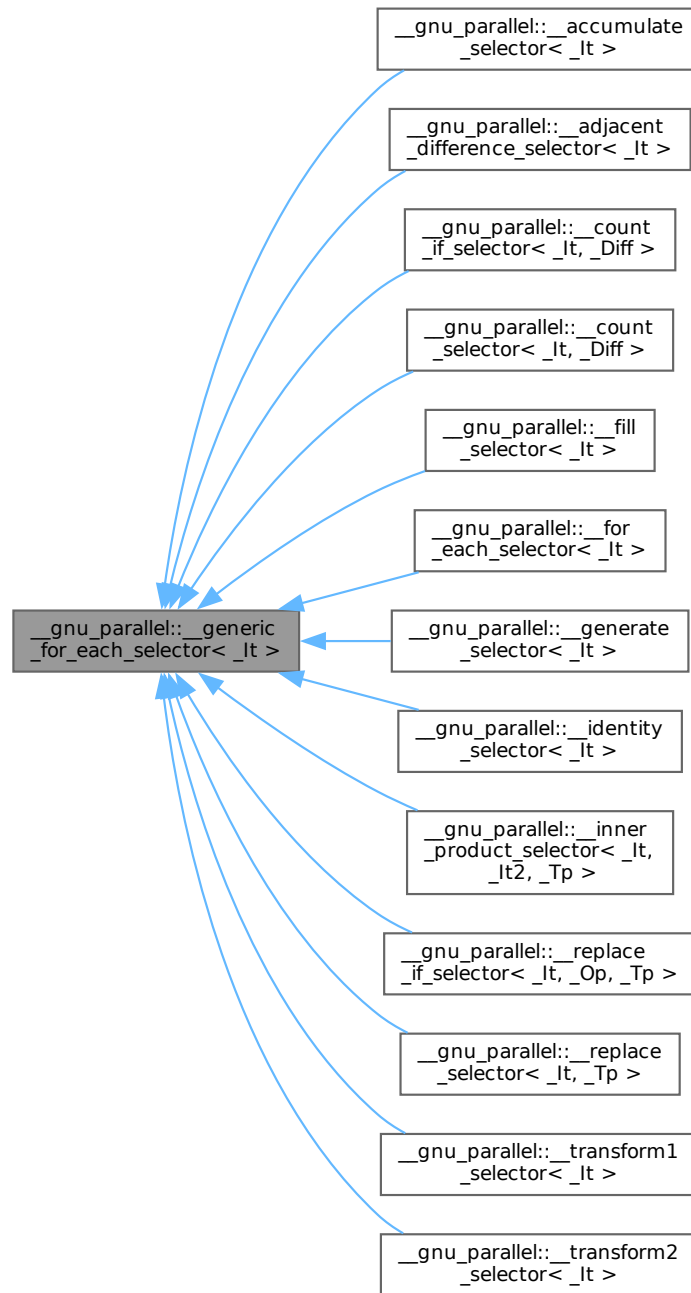
The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

6.26 __gnu_parallel::__generic_for_each_selector<_It> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__generic_for_each_selector<_It>`:



Public Attributes

- `_It` [_M_finish_iterator](#)

6.26.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__generic_for_each_selector<_It>
```

Generic `__selector` for embarrassingly parallel functions.

6.26.2 Member Data Documentation

`_M_finish_iterator`

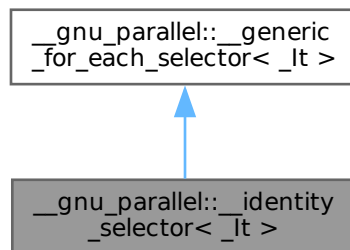
```
template<typename _It>
_It __gnu_parallel::__generic_for_each_selector<_It>::_M_finish_iterator
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
The documentation for this struct was generated from the following file:
```

- [for_each_selectors.h](#)

6.27 `__gnu_parallel::__identity_selector<_It>` Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__identity_selector<_It>`:



Public Member Functions

- `template<typename _Op>`
`_It operator() (_Op __o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.27.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__identity_selector<_It>
```

Selector that just returns the passed iterator.

6.27.2 Member Function Documentation

operator>()

```
template<typename _It >
template<typename _Op >
_It __gnu_parallel::__identity_selector< _It >::operator() (
    _Op __o,
    _It __i ) [inline]
```

Functor execution.

Parameters

\leftarrow _o	Operator (unused).
\leftarrow _i	iterator referencing object.

Returns

Passed iterator.

6.27.3 Member Data Documentation

_M_finish_iterator

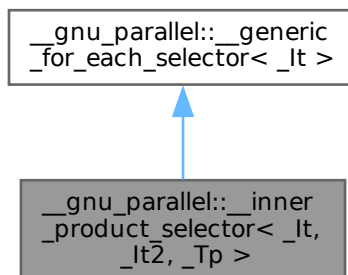
```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
The documentation for this struct was generated from the following file:
```

- [for_each_selectors.h](#)

6.28 __gnu_parallel::__inner_product_selector< _It, _It2, _Tp > Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for __gnu_parallel::__inner_product_selector< _It, _It2, _Tp >:



Public Member Functions

- [`__inner_product_selector`](#) (`_It __b1, _It2 __b2`)
- `template<typename _Op>`
`_Tp operator()` (`_Op __mult, _It __current`)

Public Attributes

- `_It __begin1_iterator`
- `_It2 __begin2_iterator`
- `_It __M_finish_iterator`

6.28.1 Detailed Description

`template<typename _It, typename _It2, typename _Tp>`
`struct __gnu_parallel::__inner_product_selector<_It, _It2, _Tp>`

`std::inner_product()` selector.

6.28.2 Constructor & Destructor Documentation

`__inner_product_selector()`

```
template<typename _It , typename _It2 , typename _Tp >
__gnu_parallel::__inner_product_selector< _It, _It2, _Tp >::__inner_product_selector (
    _It __b1,
    _It2 __b2 ) [inline], [explicit]
```

Constructor.

Parameters

<code>__b1</code>	Begin iterator of first sequence.
<code>__b2</code>	Begin iterator of second sequence.

6.28.3 Member Function Documentation

`operator>()()`

```
template<typename _It , typename _It2 , typename _Tp >
template<typename _Op >
_Tp __gnu_parallel::__inner_product_selector< _It, _It2, _Tp >::operator() (
    _Op __mult,
    _It __current ) [inline]
```

Functor execution.

Parameters

<code>__mult</code>	Multiplication functor.
<code>__current</code>	iterator referencing object.

Returns

Inner product elemental `__result`.

References [`__gnu_parallel::__inner_product_selector<_It, _It2, _Tp>::__begin1_iterator`](#), and [`__gnu_parallel::__inner_product_selector`](#)

6.28.4 Member Data Documentation

`__begin1_iterator`

```
template<typename _It , typename _It2 , typename _Tp >
_It \_\_gnu\_parallel::\_\_inner\_product\_selector< _It, _It2, _Tp >::__begin1_iterator
Begin iterator of first sequence.
Referenced by \_\_gnu\_parallel::\_\_inner\_product\_selector< _It, _It2, _Tp >::operator()().
```

`__begin2_iterator`

```
template<typename _It , typename _It2 , typename _Tp >
_It2 \_\_gnu\_parallel::\_\_inner\_product\_selector< _It, _It2, _Tp >::__begin2_iterator
Begin iterator of second sequence.
Referenced by \_\_gnu\_parallel::\_\_inner\_product\_selector< _It, _It2, _Tp >::operator()().
```

`_M_finish_iterator`

```
template<typename _It >
_It \_\_gnu\_parallel::\_\_generic\_for\_each\_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
The documentation for this struct was generated from the following file:
```

- [for_each_selectors.h](#)

6.29 `std::__is_location_invariant<_Tp>` Struct Template Reference

```
#include <std_function.h>
Inherits is_trivially_copyable::type.
```

6.29.1 Detailed Description

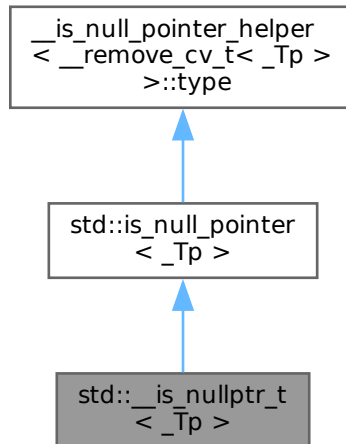
```
template<typename _Tp>
struct std::__is_location_invariant< _Tp >
```

Trait identifying "location-invariant" types, meaning that the address of the object (or any of its members) will not escape. Trivially copyable types are location-invariant and users can specialize this trait for other types. The documentation for this struct was generated from the following file:

- [std_function.h](#)

6.30 std::__is_nullptr_t< _Tp > Struct Template Reference

Inheritance diagram for std::__is_nullptr_t< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.30.1 Detailed Description

```
template<typename _Tp>
struct std::__is_nullptr_t< _Tp >
```

`__is_nullptr_t` (deprecated extension).

Deprecated Non-standard. Use `is_null_pointer` instead.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.31 __gnu_parallel::__max_element_reduct< _Compare, _It > Struct Template Reference

```
#include <for_each_selectors.h>
```

Public Member Functions

- `__max_element_reduct` (`_Compare &__c`)
- `_lt operator()` (`_lt __x, _lt __y`)

Public Attributes

- `_Compare & __comp`

6.31.1 Detailed Description

```
template<typename _Compare, typename _It>
struct __gnu_parallel::__max_element_reduct< _Compare, _It >
```

Reduction for finding the maximum element, using a comparator.
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.32 `__gnu_parallel::__min_element_reduct< _Compare, _It >` Struct Template Reference

```
#include <for_each_selectors.h>
```

Public Member Functions

- `__min_element_reduct` (`_Compare &__c`)
- `_lt operator()` (`_lt __x, _lt __y`)

Public Attributes

- `_Compare & __comp`

6.32.1 Detailed Description

```
template<typename _Compare, typename _It>
struct __gnu_parallel::__min_element_reduct< _Compare, _It >
```

Reduction for finding the maximum element, using a comparator.
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.33 `__gnu_cxx::__detail::__mini_vector< _Tp >` Class Template Reference

```
#include <bitmap_allocator.h>
```

Public Types

- `typedef const _Tp & const_reference`
- `typedef std::ptrdiff_t difference_type`
- `typedef pointer iterator`
- `typedef _Tp * pointer`
- `typedef _Tp & reference`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- reference **back** () const throw ()
- iterator **begin** () const throw ()
- void **clear** () throw ()
- iterator **end** () const throw ()
- void **erase** (iterator __pos) throw ()
- void **insert** (iterator __pos, const_reference __x)
- reference **operator[]** (const size_type __pos) const throw ()
- void **pop_back** () throw ()
- void **push_back** (const_reference __x)
- size_type **size** () const throw ()

6.33.1 Detailed Description

template<typename _Tp>
class __gnu_cxx::__detail::__mini_vector<_Tp>

__mini_vector<> is a stripped down version of the full-fledged std::vector<>. It is to be used only for built-in types or PODs. Notable differences are:

1. Not all accessor functions are present.
2. Used ONLY for PODs.
3. No Allocator template argument. Uses operator new() to get memory, and operator delete() to free it. Caveat: The dtor does NOT free the memory allocated, so this a memory-leaking vector!

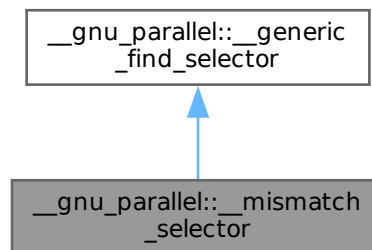
The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

6.34 __gnu_parallel::__mismatch_selector Struct Reference

```
#include <find_selectors.h>
```

Inheritance diagram for __gnu_parallel::__mismatch_selector:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair< _RAIter1, _RAIter2 > _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator() (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

6.34.1 Detailed Description

Test inverted predicate on a single element.

6.34.2 Member Function Documentation

[_M_sequential_algorithm\(\)](#)

```
template<typename _RAIter1, typename _RAIter2, typename _Pred >
std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__mismatch_selector::_M_sequential_algorithm (
    _RAIter1 __begin1,
    _RAIter1 __end1,
    _RAIter2 __begin2,
    _Pred __pred ) [inline]
```

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

[operator\(\)\(\)](#)

```
template<typename _RAIter1, typename _RAIter2, typename _Pred >
bool __gnu_parallel::__mismatch_selector::operator() (
    _RAIter1 __i1,
    _RAIter2 __i2,
    _Pred __pred ) [inline]
```

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

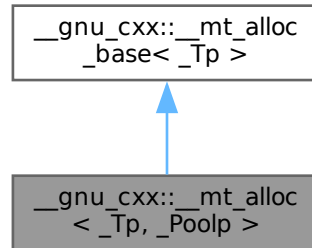
The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

6.35 `__gnu_cxx::__mt_alloc< _Tp, _Poolp >` Class Template Reference

```
#include <mt_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__mt_alloc<_Tp, _Poolp>`:



Public Types

- `typedef _Poolp __policy_type`
- `typedef _Poolp::pool_type __pool_type`
- `typedef const _Tp * const_pointer`
- `typedef const _Tp & const_reference`
- `typedef std::ptrdiff_t difference_type`
- `typedef _Tp * pointer`
- `typedef std::true_type propagate_on_container_move_assignment`
- `typedef _Tp & reference`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `__mt_alloc` (const [__mt_alloc](#) &) noexcept
- `template<typename _Tp1, typename _Poolp1 > __mt_alloc` (const [__mt_alloc](#)<_Tp1, _Poolp1 > &) noexcept
- `const __pool_base::_Tune M_get_options ()`
- `void M_set_options (__pool_base::_Tune __t)`
- `const_pointer address (const_reference __x) const noexcept`
- `pointer address (reference __x) const noexcept`
- `pointer allocate (size_type __n, const void * = 0)`
- `template<typename _Up, typename... _Args> void construct (_Up * __p, _Args &&... __args)`
- `void deallocate (pointer __p, size_type __n)`
- `template<typename _Up > void destroy (_Up * __p)`
- `size_type max_size () const noexcept`

6.35.1 Detailed Description

```
template<typename _Tp, typename _Poolp = __common_pool_policy<__pool, true >>
class __gnu_cxx::__mt_alloc< _Tp, _Poolp >
```

This is a fixed size (power of 2) allocator which - when compiled with thread support - will maintain one freelist per size per thread plus a *global* one. Steps are taken to limit the per thread freelist sizes (by returning excess back to the *global* list).

Further details: https://gcc.gnu.org/onlinedocs/libstdc++/manual/mt_allocator.html

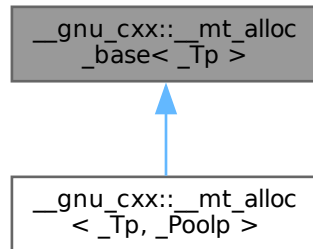
The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

6.36 __gnu_cxx::__mt_alloc_base< _Tp > Class Template Reference

```
#include <mt_allocator.h>
```

Inheritance diagram for __gnu_cxx::__mt_alloc_base< _Tp >:



Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef std::ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- const_pointer **address** (const_reference __x) const noexcept
- pointer **address** (reference __x) const noexcept
- template<typename _Up, typename... _Args>
void **construct** (_Up *__p, _Args &&... __args)
- template<typename _Up >
void **destroy** (_Up *__p)
- size_type **max_size** () const noexcept

6.36.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__mt_alloc_base< _Tp >
```

Base class for `_Tp` dependent member functions.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

6.37 `__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, _↵ DifferenceTp __length, _Compare __comp)`

6.37.1 Detailed Description

```
template<bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for 3-way merging with `__sentinels` turned off.

Note that 3-way merging is always stable!

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.38 `__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, _↵ DifferenceTp __length, _Compare __comp)`

6.38.1 Detailed Description

```
template<typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _↵ DifferenceTp, _Compare >
```

Switch for 3-way merging with `__sentinels` turned on.

Note that 3-way merging is always stable!

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.39 `__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator()` (`_RAIterlterator __seqs_begin`, `_RAIterlterator __seqs_end`, `_RAIter3 __target`, `__DifferenceTp __length`, `_Compare __comp`)

6.39.1 Detailed Description

```
template<bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
```

```
struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for 4-way merging with `__sentinels` turned off.

Note that 4-way merging is always stable!

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.40 `__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator()` (`_RAIterlterator __seqs_begin`, `_RAIterlterator __seqs_end`, `_RAIter3 __target`, `__DifferenceTp __length`, `_Compare __comp`)

6.40.1 Detailed Description

```
template<typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
```

```
struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, __DifferenceTp, _Compare >
```

Switch for 4-way merging with `__sentinels` turned on.

Note that 4-way merging is always stable!

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.41 `__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator()` (`_RAIterlterator __seqs_begin`, `_RAIterlterator __seqs_end`, `_RAIter3 __target`, `const typename std::iterator_traits< typename std::iterator_traits< _RAIterlterator >::value_type::first_type >::value_type & __sentinel`, `_DifferenceTp __length`, `_Compare __comp`)

6.41.1 Detailed Description

```
template<bool __sentinels, bool __stable, typename _RAIterlterator, typename _RAIter3, typename _↵
_DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterlterator,
_RAlter3, _DifferenceTp, _Compare >
```

Switch for k-way merging with `__sentinels` turned on.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.42 `__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, const type-
name std::iterator_traits< typename std::iterator_traits< _RAIterlterator >::value_type::first_type >::value_type
&__sentinel, _DifferenceTp __length, _Compare __comp)`

6.42.1 Detailed Description

```
template<bool __stable, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename
_Compare>
struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterlterator, _↵
_RAlter3, _DifferenceTp, _Compare >
```

Switch for k-way merging with `__sentinels` turned off.

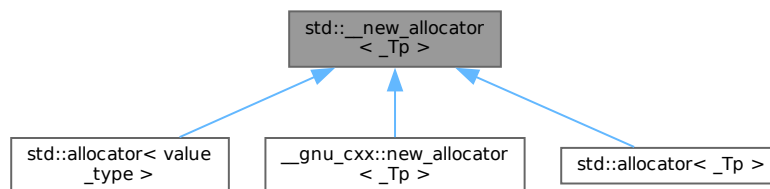
The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.43 `std::__new_allocator< _Tp >` Class Template Reference

```
#include <new_allocator.h>
```

Inheritance diagram for `std::__new_allocator< _Tp >`:



Public Types

- `typedef std::ptrdiff_t difference_type`
- `typedef std::true_type propagate_on_container_move_assignment`

- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **__new_allocator** (const [__new_allocator](#) &) noexcept
- template<typename _Tp1 >
constexpr **__new_allocator** (const [__new_allocator](#)<_Tp1 > &) noexcept
- _Tp * **allocate** (size_type __n, const void * =static_cast< const void * >(0))
- void **deallocate** (_Tp * __p, size_type __n)

Friends

- template<typename _Up >
constexpr bool **operator==** (const [__new_allocator](#) &, const [__new_allocator](#)<_Up > &) noexcept

6.43.1 Detailed Description

template<typename _Tp>
class std::__new_allocator<_Tp >

An allocator that uses global new, as per C++03 [20.4.1].
 This is precisely the allocator defined in the C++ Standard.

- all allocation calls operator new
- all deallocation calls operator delete

Template Parameters

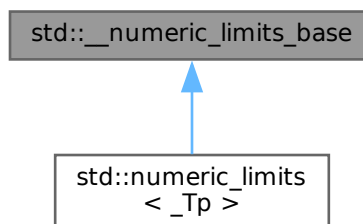
<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

The documentation for this class was generated from the following file:

- [bits/new_allocator.h](#)

6.44 std::__numeric_limits_base Struct Reference

Inheritance diagram for std::__numeric_limits_base:



Static Public Attributes

- static constexpr int [digits](#)
- static constexpr int [digits10](#)
- static constexpr [float_denorm_style](#) [has_denorm](#)
- static constexpr bool [has_denorm_loss](#)
- static constexpr bool [has_infinity](#)
- static constexpr bool [has_quiet_NaN](#)
- static constexpr bool [has_signaling_NaN](#)
- static constexpr bool [is_bounded](#)
- static constexpr bool [is_exact](#)
- static constexpr bool [is_iec559](#)
- static constexpr bool [is_integer](#)
- static constexpr bool [is_modulo](#)
- static constexpr bool [is_signed](#)
- static constexpr bool [is_specialized](#)
- static constexpr int [max_digits10](#)
- static constexpr int [max_exponent](#)
- static constexpr int [max_exponent10](#)
- static constexpr int [min_exponent](#)
- static constexpr int [min_exponent10](#)
- static constexpr int [radix](#)
- static constexpr [float_round_style](#) [round_style](#)
- static constexpr bool [tinyness_before](#)
- static constexpr bool [traps](#)

6.44.1 Detailed Description

Part of std::numeric_limits.

The `static const` members are usable as integral constant expressions.

Note

This is a separate class for purposes of efficiency; you should only access these members as part of an instantiation of the std::numeric_limits class.

6.44.2 Member Data Documentation

digits

```
constexpr int std::__numeric_limits_base::digits [static], [constexpr]
```

The number of `radix` digits that be represented without change: for integer types, the number of non-sign bits in the mantissa; for floating types, the number of `radix` digits in the mantissa.

digits10

```
constexpr int std::__numeric_limits_base::digits10 [static], [constexpr]
```

The number of base 10 digits that can be represented without change.

has_denorm

```
constexpr float\_denorm\_style std::__numeric_limits_base::has_denorm [static], [constexpr]
```

See std::float_denorm_style for more information.

has_denorm_loss

`constexpr bool std::__numeric_limits_base::has_denorm_loss [static], [constexpr]`
True if loss of accuracy is detected as a denormalization loss, rather than as an inexact result.

has_infinity

`constexpr bool std::__numeric_limits_base::has_infinity [static], [constexpr]`
True if the type has a representation for positive infinity.

has_quiet_NaN

`constexpr bool std::__numeric_limits_base::has_quiet_NaN [static], [constexpr]`
True if the type has a representation for a quiet (non-signaling) Not a Number.

has_signaling_NaN

`constexpr bool std::__numeric_limits_base::has_signaling_NaN [static], [constexpr]`
True if the type has a representation for a signaling Not a Number.

is_bounded

`constexpr bool std::__numeric_limits_base::is_bounded [static], [constexpr]`
True if the set of values representable by the type is finite. All built-in types are bounded, this member would be false for arbitrary precision types.

is_exact

`constexpr bool std::__numeric_limits_base::is_exact [static], [constexpr]`
True if the type uses an exact representation. All integer types are exact, but not all exact types are integer. For example, rational and fixed-exponent representations are exact but not integer.

is_iec559

`constexpr bool std::__numeric_limits_base::is_iec559 [static], [constexpr]`
True if-and-only-if the type adheres to the IEC 559 standard, also known as IEEE 754. (Only makes sense for floating point types.)

is_integer

`constexpr bool std::__numeric_limits_base::is_integer [static], [constexpr]`
True if the type is integer.

is_modulo

`constexpr bool std::__numeric_limits_base::is_modulo [static], [constexpr]`
True if the type is *modulo*. A type is modulo if, for any operation involving +, -, or * on values of that type whose result would fall outside the range [min(),max()), the value returned differs from the true value by an integer multiple of max() - min() + 1. On most machines, this is false for floating types, true for unsigned integers, and true for signed integers. See PR22200 about signed integers.

is_signed

```
constexpr bool std::__numeric_limits_base::is_signed [static], [constexpr]
```

True if the type is signed.

is_specialized

```
constexpr bool std::__numeric_limits_base::is_specialized [static], [constexpr]
```

This will be true for all fundamental types (which have specializations), and false for everything else.

max_digits10

```
constexpr int std::__numeric_limits_base::max_digits10 [static], [constexpr]
```

The number of base 10 digits required to ensure that values which differ are always differentiated.

max_exponent

```
constexpr int std::__numeric_limits_base::max_exponent [static], [constexpr]
```

The maximum positive integer such that `radix` raised to the power of (one less than that integer) is a representable finite floating point number.

max_exponent10

```
constexpr int std::__numeric_limits_base::max_exponent10 [static], [constexpr]
```

The maximum positive integer such that 10 raised to that power is in the range of representable finite floating point numbers.

min_exponent

```
constexpr int std::__numeric_limits_base::min_exponent [static], [constexpr]
```

The minimum negative integer such that `radix` raised to the power of (one less than that integer) is a normalized floating point number.

min_exponent10

```
constexpr int std::__numeric_limits_base::min_exponent10 [static], [constexpr]
```

The minimum negative integer such that 10 raised to that power is in the range of normalized floating point numbers.

radix

```
constexpr int std::__numeric_limits_base::radix [static], [constexpr]
```

For integer types, specifies the base of the representation. For floating types, specifies the base of the exponent representation.

round_style

```
constexpr float_round_style std::__numeric_limits_base::round_style [static], [constexpr]
```

See `std::float_round_style` for more information. This is only meaningful for floating types; integer types will all be `round_toward_zero`.

tinyness_before

`constexpr bool std::__numeric_limits_base::tinyness_before [static], [constexpr]`
True if tininess is detected before rounding. (see IEC 559)

traps

`constexpr bool std::__numeric_limits_base::traps [static], [constexpr]`
True if trapping is implemented for this type.

The documentation for this struct was generated from the following file:

- [limits](#)

6.45 `__gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread >` Struct Template Reference

```
#include <mt_allocator.h>
Inherits __gnu_cxx::__per_type_pool_base<_Tp, _PoolTp, _Thread >.
```

6.45.1 Detailed Description

```
template<typename _Tp, template< bool > class _PoolTp, bool _Thread>
struct __gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread >
```

Policy for individual `__pool` objects.

The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

6.46 `__gnu_cxx::__pool<_Thread >` Class Template Reference

6.46.1 Detailed Description

```
template<bool _Thread>
class __gnu_cxx::__pool<_Thread >
```

Data describing the underlying memory pool, parameterized on threading support.

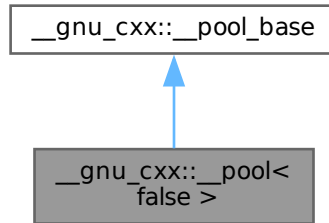
The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

6.47 `__gnu_cxx::__pool<false >` Class Reference

```
#include <mt_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__pool<false>`:



Public Types

- typedef unsigned short int **_Binmap_type**
- typedef std::size_t **size_t**

Public Member Functions

- **__pool** (const __pool_base::_Tune &__tune)
- void **_M_adjust_freelist** (const _Bin_record &, _Block_record *, size_t)
- bool **_M_check_threshold** (size_t __bytes)
- void **_M_destroy** () throw ()
- size_t **_M_get_align** ()
- const _Bin_record & **_M_get_bin** (size_t __which)
- size_t **_M_get_binmap** (size_t __bytes)
- const _Tune & **_M_get_options** () const
- size_t **_M_get_thread_id** ()
- void **_M_initialize_once** ()
- void **_M_reclaim_block** (char *__p, size_t __bytes) throw ()
- char * **_M_reserve_block** (size_t __bytes, const size_t __thread_id)
- void **_M_set_options** (_Tune __t)

Protected Attributes

- _Binmap_type * **_M_binmap**
- bool **_M_init**
- _Tune **_M_options**

6.47.1 Detailed Description

Specialization for single thread.

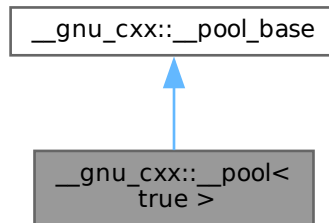
The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

6.48 `__gnu_cxx::__pool< true >` Class Reference

```
#include <mt_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__pool< true >`:



Public Types

- typedef unsigned short int **_Binmap_type**
- typedef std::size_t **size_t**

Public Member Functions

- **__pool** (const __pool_base::Tune & __tune)
- void **_M_adjust_freelist** (const _Bin_record & __bin, _Block_record * __block, size_t __thread_id)
- bool **_M_check_threshold** (size_t __bytes)
- void **_M_destroy** () throw ()
- void **_M_destroy_thread_key** (void *) throw ()
- size_t **_M_get_align** ()
- const _Bin_record & **_M_get_bin** (size_t __which)
- size_t **_M_get_binmap** (size_t __bytes)
- const Tune & **_M_get_options** () const
- size_t **_M_get_thread_id** ()
- void **_M_initialize** (__destroy_handler)
- void **_M_initialize_once** ()
- void **_M_reclaim_block** (char * __p, size_t __bytes) throw ()
- char * **_M_reserve_block** (size_t __bytes, const size_t __thread_id)
- void **_M_set_options** (_Tune __t)

Protected Attributes

- _Binmap_type * **_M_binmap**
- bool **_M_init**
- _Tune **_M_options**

6.48.1 Detailed Description

Specialization for thread enabled, via gthreads.h.

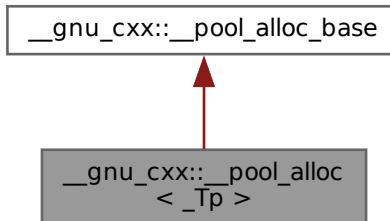
The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

6.49 `__gnu_cxx::__pool_alloc<_Tp>` Class Template Reference

```
#include <pool_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__pool_alloc<_Tp>`:



Public Types

- typedef const `_Tp` * **const_pointer**
- typedef const `_Tp` & **const_reference**
- typedef std::ptrdiff_t **difference_type**
- typedef `_Tp` * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef `_Tp` & **reference**
- typedef std::size_t **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `__pool_alloc` (const [__pool_alloc](#) &) noexcept
- template<typename `_Tp1` >
 [__pool_alloc](#) (const [__pool_alloc](#)< `_Tp1` > &) noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **address** (reference __x) const noexcept
- pointer **allocate** (size_type __n, const void * = 0)
- template<typename `_Up`, typename... `_Args`>
 void **construct** (`_Up` * __p, `_Args` &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- template<typename `_Up` >
 void **destroy** (`_Up` * __p)
- size_type **max_size** () const noexcept

6.49.1 Detailed Description

```
template<typename _Tp>
```

```
class __gnu_cxx::__pool_alloc<_Tp>
```

Allocator using a memory pool with a single lock.

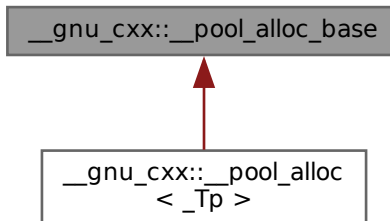
The documentation for this class was generated from the following file:

- [pool_allocator.h](#)

6.50 `__gnu_cxx::__pool_alloc_base` Class Reference

```
#include <pool_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__pool_alloc_base`:



Protected Types

- enum { `_S_align` }
- enum { `_S_max_bytes` }
- enum { `_S_free_list_size` }

Protected Member Functions

- `char * _M_allocate_chunk (size_t __n, int &__nobjs)`
- `_Obj *volatile * _M_get_free_list (size_t __bytes) throw ()`
- `__mutex & _M_get_mutex () throw ()`
- `void * _M_refill (size_t __n)`
- `size_t _M_round_up (size_t __bytes)`

Static Protected Attributes

- static `char * _S_end_free`
- static `_Obj *volatile _S_free_list [_S_free_list_size]`
- static `size_t _S_heap_size`
- static `char * _S_start_free`

6.50.1 Detailed Description

Base class for `__pool_alloc`.

Uses various allocators to fulfill underlying requests (and makes as few requests as possible when in default high-speed pool mode).

Important implementation properties: 0. If globally mandated, then allocate objects from new

1. If the clients request an object of size $> _S_max_bytes$, the resulting object will be obtained directly from new
2. In all other cases, we allocate an object of size exactly `__S_round_up(requested_size)`. Thus the client has enough size information that we can return the object to the proper free list without permanently losing part of the object.

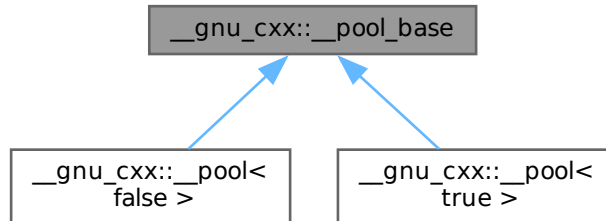
The documentation for this class was generated from the following file:

- [pool_allocator.h](#)

6.51 `__gnu_cxx::__pool_base` Struct Reference

```
#include <mt_allocator.h>
```

Inheritance diagram for `__gnu_cxx::__pool_base`:



Public Types

- typedef unsigned short int **_Binmap_type**
- typedef std::size_t **size_t**

Public Member Functions

- **__pool_base** (const _Tune &__options)
- bool **_M_check_threshold** (size_t __bytes)
- size_t **_M_get_align** ()
- size_t **_M_get_binmap** (size_t __bytes)
- const _Tune & **_M_get_options** () const
- void **_M_set_options** (_Tune __t)

Protected Attributes

- _Binmap_type * **_M_binmap**
- bool **_M_init**
- _Tune **_M_options**

6.51.1 Detailed Description

Base class for pool object.

The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

6.52 `__gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>` Class Template Reference

```
#include <rc_string_base.h>
```

Inherits `__gnu_cxx::__vstring_utility<_CharT, _Traits, _Alloc>`.

Public Types

- typedef `_Util_Base::_CharT_alloc_type` **`_CharT_alloc_type`**
- typedef `__vstring_utility<_CharT, _Traits, _Alloc>` **`_Util_Base`**
- typedef `_Alloc` **`allocator_type`**
- typedef `_CharT_alloc_type::size_type` **`size_type`**
- typedef `_Traits` **`traits_type`**
- typedef `_Traits::char_type` **`value_type`**

Public Member Functions

- `__rc_string_base` (`__rc_string_base` && __rcs)
- template<typename `_InputIterator` >
`__rc_string_base` (`_InputIterator` __beg, `_InputIterator` __end, const `_Alloc` & __a)
- `__rc_string_base` (const `__rc_string_base` & __rcs)
- `__rc_string_base` (const `_Alloc` & __a)
- `__rc_string_base` (size_type __n, `_CharT` __c, const `_Alloc` & __a)
- void `_M_assign` (const `__rc_string_base` & __rcs)
- size_type `_M_capacity` () const
- void `_M_clear` ()
- bool `_M_compare` (const `__rc_string_base` &) const
- bool `_M_compare` (const `__rc_string_base` & __rcs) const
- bool `_M_compare` (const `__rc_string_base` & __rcs) const
- `_CharT` * `_M_data` () const
- void `_M_erase` (size_type __pos, size_type __n)
- allocator_type & `_M_get_allocator` ()
- const allocator_type & `_M_get_allocator` () const
- bool `_M_is_shared` () const
- void `_M_leak` ()
- size_type `_M_length` () const
- size_type `_M_max_size` () const
- void `_M_mutate` (size_type __pos, size_type __len1, const `_CharT` * __s, size_type __len2)
- void `_M_reserve` (size_type __res)
- void `_M_set_leaked` ()
- void `_M_set_length` (size_type __n)
- void `_M_swap` (`__rc_string_base` & __rcs)
- template<typename `_InIterator` >
`_CharT` * `_S_construct` (`_InIterator` __beg, `_InIterator` __end, const `_Alloc` & __a, `std::forward_iterator_tag`)

Protected Types

- typedef `__gnu_cxx::__normal_iterator< const_pointer, __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, __rc_string_base>>` **`__const_rc_iterator`**
- typedef `__gnu_cxx::__normal_iterator< const_pointer, __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, __sso_string_base>>` **`__const_sso_iterator`**
- typedef `__gnu_cxx::__normal_iterator< pointer, __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, __rc_string_base>>` **`__rc_iterator`**
- typedef `__gnu_cxx::__normal_iterator< pointer, __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, __sso_string_base>>` **`__sso_iterator`**
- typedef `__alloc_traits<_CharT_alloc_type>` **`_CharT_alloc_traits`**
- typedef `_CharT_alloc_traits::const_pointer` **`const_pointer`**
- typedef `_CharT_alloc_type::difference_type` **`difference_type`**
- typedef `_CharT_alloc_traits::pointer` **`pointer`**

Static Protected Member Functions

- static void **_S_assign** (_CharT *__d, size_type __n, _CharT __c)
- static int **_S_compare** (size_type __n1, size_type __n2)
- static void **_S_copy** (_CharT *__d, const _CharT *__s, size_type __n)
- static void **_S_copy_chars** (_CharT *__p, __const_rc_iterator __k1, __const_rc_iterator __k2)
- static void **_S_copy_chars** (_CharT *__p, __const_sso_iterator __k1, __const_sso_iterator __k2)
- static void **_S_copy_chars** (_CharT *__p, __rc_iterator __k1, __rc_iterator __k2)
- static void **_S_copy_chars** (_CharT *__p, __sso_iterator __k1, __sso_iterator __k2)
- static void **_S_copy_chars** (_CharT *__p, _CharT *__k1, _CharT *__k2)
- template<typename _Iterator>
static void **_S_copy_chars** (_CharT *__p, _Iterator __k1, _Iterator __k2)
- static void **_S_copy_chars** (_CharT *__p, const _CharT *__k1, const _CharT *__k2)
- static void **_S_move** (_CharT *__d, const _CharT *__s, size_type __n)

6.52.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class __gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>
```

Documentation? What's that? Nathan Myers ncm@cantrip.org.

A string looks like this:

```

                                     [_Rep]
                                     _M_length
[ __rc_string_base<char_type>]      _M_capacity
_M_datapointer                    _M_refcount
_M_p ----->                     unnamed array of char_type
```

Where the `_M_p` points to the first character in the string, and you cast it to a pointer-to-`_Rep` and subtract 1 to get a pointer to the header.

This approach has the enormous advantage that a string object requires only one allocation. All the ugliness is confined within a single pair of inline functions, which each compile to a single *add* instruction: `_Rep::_M_refdata()`, and `__rc_string_base::_M_rep()`; and the allocation function which gets a block of raw bytes and with room enough and constructs a `_Rep` object at the front.

The reason you want `_M_data` pointing to the character array and not the `_Rep` is so that the debugger can see the string contents. (Probably we should add a non-inline member to get the `_Rep` for the debugger to use, so users can check the actual string length.)

Note that the `_Rep` object is a POD so that you can have a static *empty string* `_Rep` object already *constructed* before static constructors have run. The reference-count encoding is chosen so that a 0 indicates one reference, so you never try to destroy the empty-string `_Rep` object.

All but the last paragraph is considered pretty conventional for a C++ string implementation.

The documentation for this class was generated from the following file:

- [rc_string_base.h](#)

6.53 std::tr2::__reflection_typelist<_Elements> Struct Template Reference

6.53.1 Detailed Description

```
template<typename... _Elements>
struct std::tr2::__reflection_typelist<_Elements>
```

See N2965: Type traits and base classes by Michael Spertus Simple typelist. Compile-time list of types.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

6.54 `std::tr2::__reflection_typelist< _First, _Rest... >` Struct Template Reference

Public Types

- typedef `std::false_type` `empty`

6.54.1 Detailed Description

```
template<typename _First, typename... _Rest>
struct std::tr2::__reflection_typelist< _First, _Rest... >
```

Partial specialization.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

6.55 `std::tr2::__reflection_typelist<>` Struct Reference

Public Types

- typedef `std::true_type` `empty`

6.55.1 Detailed Description

Specialization for an empty typelist.

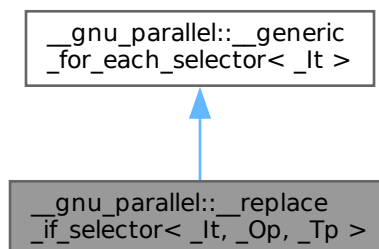
The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

6.56 `__gnu_parallel::__replace_if_selector< _It, _Op, _Tp >` Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__replace_if_selector< _It, _Op, _Tp >`:



Public Member Functions

- `__replace_if_selector` (const `_Tp` & `__new_val`)
- bool `operator()` (`_Op` & `__o`, `_It` `__i`)

Public Attributes

- `const _Tp & __new_val`
- `_It __M_finish_iterator`

6.56.1 Detailed Description

```
template<typename _It, typename _Op, typename _Tp>
struct __gnu_parallel::__replace_if_selector<_It, _Op, _Tp>
```

`std::replace()` selector.

6.56.2 Constructor & Destructor Documentation**`__replace_if_selector()`**

```
template<typename _It , typename _Op , typename _Tp >
__gnu_parallel::__replace_if_selector< _It, _Op, _Tp >::__replace_if_selector (
    const _Tp & __new_val ) [inline], [explicit]
```

Constructor.

Parameters

<code>__new_val</code>	Value to replace with.
------------------------	------------------------

6.56.3 Member Function Documentation**`operator>()()`**

```
template<typename _It , typename _Op , typename _Tp >
bool __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >::operator() (
    _Op & __o,
    _It __i ) [inline]
```

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

References `__gnu_parallel::__replace_if_selector<_It, _Op, _Tp>::__new_val`.

6.56.4 Member Data Documentation**`__new_val`**

```
template<typename _It , typename _Op , typename _Tp >
const _Tp& __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >::__new_val
Value to replace with.
Referenced by __gnu_parallel::__replace_if_selector<_It, _Op, _Tp>::operator>()().
```

`__M_finish_iterator`

```
template<typename _It >
```


`_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator` [inherited]
`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

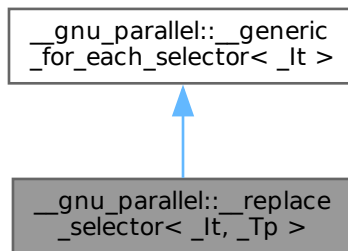
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.57 __gnu_parallel::__replace_selector< _It, _Tp > Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__replace_selector< _It, _Tp >`:



Public Member Functions

- `__replace_selector` (const `_Tp` & `__new_val`)
- bool `operator()` (`_Tp` & `__v`, `_It` `__i`)

Public Attributes

- const `_Tp` & `__new_val`
- `_It` `_M_finish_iterator`

6.57.1 Detailed Description

```
template<typename _It, typename _Tp>
struct __gnu_parallel::__replace_selector< _It, _Tp >
```

`std::replace()` selector.

6.57.2 Constructor & Destructor Documentation

`__replace_selector()`

```
template<typename _It , typename _Tp >
__gnu_parallel::__replace_selector< _It, _Tp >::__replace_selector (
    const _Tp & __new_val ) [inline], [explicit]
```

Constructor.

Parameters

<code>__new_val</code>	Value to replace with.
------------------------	------------------------

6.57.3 Member Function Documentation

`operator()()`

```
template<typename _It , typename _Tp >
bool __gnu_parallel::__replace_selector< _It, _Tp >::operator() (
    _Tp & __v,
    _It __i ) [inline]
```

Functor execution.

Parameters

<code>__v</code>	Current value.
<code>__i</code>	iterator referencing object.

References [__gnu_parallel::__replace_selector< _It, _Tp >::__new_val](#).

6.57.4 Member Data Documentation

`__new_val`

```
template<typename _It , typename _Tp >
const _Tp& __gnu_parallel::__replace_selector< _It, _Tp >::__new_val
```

Value to replace with.

Referenced by [__gnu_parallel::__replace_selector< _It, _Tp >::operator\(\)\(\)](#).

`_M_finish_iterator`

```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.58 `__gnu_cxx::__scoped_lock` Class Reference

```
#include <concurrency.h>
```

Public Types

- typedef `__mutex` `__mutex_type`

Public Member Functions

- `__scoped_lock` (`__mutex_type` & `__name`)

6.58.1 Detailed Description

Scoped lock idiom.

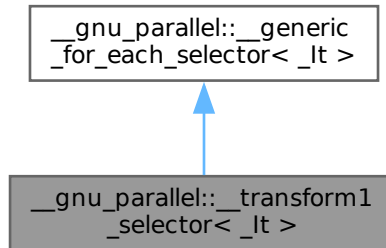
The documentation for this class was generated from the following file:

- [concurrency.h](#)

6.59 __gnu_parallel::__transform1_selector<_It> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for __gnu_parallel::__transform1_selector<_It>:



Public Member Functions

- template<typename _Op >
bool [operator\(\)](#) (_Op &__o, _It __i)

Public Attributes

- _It [_M_finish_iterator](#)

6.59.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__transform1_selector<_It>
```

std::transform() __selector, one input sequence variant.

6.59.2 Member Function Documentation

operator>()

```
template<typename _It >
template<typename _Op >
bool __gnu_parallel::__transform1_selector<_It>::operator() (
    _Op & __o,
    _It __i ) [inline]
```

Functor execution.

Parameters

__o	Operator.
__i	iterator referencing object.

6.59.3 Member Data Documentation

`_M_finish_iterator`

```
template<typename _It >
```

```
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]  
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

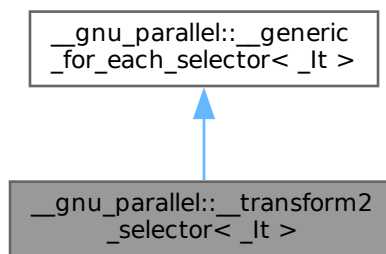
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.60 __gnu_parallel::__transform2_selector<_It> Struct Template Reference

```
#include <for_each_selectors.h>
```

Inheritance diagram for `__gnu_parallel::__transform2_selector<_It>`:



Public Member Functions

- `template<typename _Op >
bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

6.60.1 Detailed Description

```
template<typename _It>  
struct __gnu_parallel::__transform2_selector< _It >
```

`std::transform()` __selector, two input sequences variant.

6.60.2 Member Function Documentation

`operator()()`

```
template<typename _It >  
template<typename _Op >  
bool __gnu_parallel::__transform2_selector< _It >::operator() (  
    _Op & __o,  
    _It __i ) [inline]
```

Functor execution.

Parameters

<code>_o</code>	Operator.
<code>_i</code>	iterator referencing object.

6.60.3 Member Data Documentation

`_M_finish_iterator`

```
template<typename _It >
_It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator [inherited]
_iterator on last element processed; needed for some algorithms (e. g. std::transform()).
```

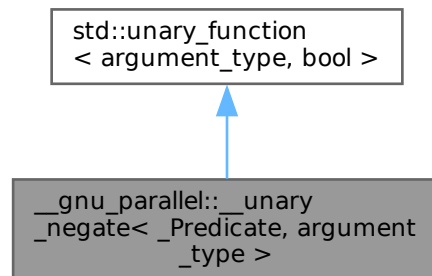
The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.61 `__gnu_parallel::__unary_negate< _Predicate, argument_type >` Class Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::__unary_negate< _Predicate, argument_type >`:



Public Types

- typedef [argument_type](#) `argument_type`
- typedef bool [result_type](#)

Public Member Functions

- `__unary_negate` (const `_Predicate` &`__x`)
- bool `operator()` (const [argument_type](#) &`__x`)

Protected Attributes

- `_Predicate _M_pred`

6.61.1 Detailed Description

```
template<typename _Predicate, typename argument\_type>
class __gnu_parallel::__unary_negate< _Predicate, argument\_type >
```

Similar to `std::unary_negate`, but giving the argument types explicitly.

6.61.2 Member Typedef Documentation

`argument_type`

```
typedef argument\_type std::unary_function< argument\_type , bool >::argument_type [inherited]
argument\_type is the type of the argument
```

`result_type`

```
typedef bool std::unary_function< argument\_type , bool >::result_type [inherited]
result\_type is the return type
```

The documentation for this class was generated from the following file:

- [base.h](#)

6.62 `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>` Class Template Reference

```
#include <vstring.h>
Inherits _Base<_CharT, _Traits, _Alloc>.
```

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `__gnu_cxx::__normal_iterator< const_pointer, __versa_string >` **const_iterator**
- typedef `_CharT_alloc_traits::const_pointer` **const_pointer**
- typedef `const value_type &` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_CharT_alloc_type::difference_type` **difference_type**
- typedef `__gnu_cxx::__normal_iterator< pointer, __versa_string >` **iterator**
- typedef `_CharT_alloc_traits::pointer` **pointer**
- typedef `value_type &` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_CharT_alloc_type::size_type` **size_type**
- typedef `_Traits` **traits_type**
- typedef `_Traits::char_type` **value_type**

Public Member Functions

- `__versa_string (__versa_string &&__str) noexcept`
- `template<class _InputIterator, typename = std::_RequireInputIter<_InputIterator>>
__versa_string (_InputIterator __beg, _InputIterator __end, const _Alloc &__a=_Alloc())`
- `__versa_string (const __versa_string &__str)`
- `__versa_string (const __versa_string &__str, size_type __pos, size_type __n, const _Alloc &__a)`

- [__versa_string](#) (const [__versa_string](#) &__str, size_type __pos, size_type __n=[npos](#))
- [__versa_string](#) (const [_Alloc](#) &__a=[_Alloc\(\)](#)) noexcept
- [__versa_string](#) (const [_CharT](#) *__s, const [_Alloc](#) &__a=[_Alloc\(\)](#))
- [__versa_string](#) (const [_CharT](#) *__s, size_type __n, const [_Alloc](#) &__a=[_Alloc\(\)](#))
- [__versa_string](#) (size_type __n, [_CharT](#) __c, const [_Alloc](#) &__a=[_Alloc\(\)](#))
- [__versa_string](#) ([std::initializer_list](#)< [_CharT](#) > __l, const [_Alloc](#) &__a=[_Alloc\(\)](#))
- [~__versa_string](#) () noexcept
- template<typename [_InputIterator](#) >
[__versa_string](#)< [_CharT](#), [_Traits](#), [_Alloc](#), [_Base](#) > & [M_replace_dispatch](#) (const_iterator __i1, const_iterator __i2, [_InputIterator](#) __k1, [_InputIterator](#) __k2, [std::false_type](#))
- template<class [_InputIterator](#) , typename = [std::RequireInputIter](#)< [_InputIterator](#)>>
[__versa_string](#) & [append](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [__versa_string](#) & [append](#) (const [__versa_string](#) &__str)
- [__versa_string](#) & [append](#) (const [__versa_string](#) &__str, size_type __pos, size_type __n)
- [__versa_string](#) & [append](#) (const [_CharT](#) *__s)
- [__versa_string](#) & [append](#) (const [_CharT](#) *__s, size_type __n)
- [__versa_string](#) & [append](#) (size_type __n, [_CharT](#) __c)
- [__versa_string](#) & [append](#) ([std::initializer_list](#)< [_CharT](#) > __l)
- [__versa_string](#) & [assign](#) ([__versa_string](#) &&__str) noexcept
- template<class [_InputIterator](#) , typename = [std::RequireInputIter](#)< [_InputIterator](#)>>
[__versa_string](#) & [assign](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [__versa_string](#) & [assign](#) (const [__versa_string](#) &__str)
- [__versa_string](#) & [assign](#) (const [__versa_string](#) &__str, size_type __pos, size_type __n)
- [__versa_string](#) & [assign](#) (const [_CharT](#) *__s)
- [__versa_string](#) & [assign](#) (const [_CharT](#) *__s, size_type __n)
- [__versa_string](#) & [assign](#) (size_type __n, [_CharT](#) __c)
- [__versa_string](#) & [assign](#) ([std::initializer_list](#)< [_CharT](#) > __l)
- reference [at](#) (size_type __n)
- const_reference [at](#) (size_type __n) const
- const_reference [back](#) () const noexcept
- reference [back](#) () noexcept
- const_iterator [begin](#) () const noexcept
- iterator [begin](#) () noexcept
- const [_CharT](#) * [c_str](#) () const noexcept
- size_type [capacity](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) () noexcept
- int [compare](#) (const [__versa_string](#) &__str) const
- int [compare](#) (const [_CharT](#) *__s) const
- int [compare](#) (size_type __pos, size_type __n, const [__versa_string](#) &__str) const
- int [compare](#) (size_type __pos, size_type __n1, const [_CharT](#) *__s) const
- int [compare](#) (size_type __pos, size_type __n1, const [_CharT](#) *__s, size_type __n2) const
- int [compare](#) (size_type __pos1, size_type __n1, const [__versa_string](#) &__str, size_type __pos2, size_type __n2) const
- size_type [copy](#) ([_CharT](#) *__s, size_type __n, size_type __pos=0) const
- const_reverse_iterator [crbegin](#) () const noexcept
- const_reverse_iterator [crend](#) () const noexcept
- const [_CharT](#) * [data](#) () const noexcept
- bool [empty](#) () const noexcept
- const_iterator [end](#) () const noexcept

- iterator `end` () noexcept
- iterator `erase` (const_iterator __first, const_iterator __last)
- iterator `erase` (const_iterator __position)
- `__versa_string` & `erase` (size_type __pos=0, size_type __n=npos)
- size_type `find` (_CharT __c, size_type __pos=0) const noexcept
- size_type `find` (const `__versa_string` & __str, size_type __pos=0) const noexcept
- size_type `find` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `find` (const _CharT * __s, size_type __pos=0) const
- size_type `find_first_not_of` (_CharT __c, size_type __pos=0) const noexcept
- size_type `find_first_not_of` (const `__versa_string` & __str, size_type __pos=0) const noexcept
- size_type `find_first_not_of` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `find_first_not_of` (const _CharT * __s, size_type __pos=0) const
- size_type `find_first_of` (_CharT __c, size_type __pos=0) const noexcept
- size_type `find_first_of` (const `__versa_string` & __str, size_type __pos=0) const noexcept
- size_type `find_first_of` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `find_first_of` (const _CharT * __s, size_type __pos=0) const
- size_type `find_last_not_of` (_CharT __c, size_type __pos=npo) const noexcept
- size_type `find_last_not_of` (const `__versa_string` & __str, size_type __pos=npo) const noexcept
- size_type `find_last_not_of` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `find_last_not_of` (const _CharT * __s, size_type __pos=npo) const
- size_type `find_last_of` (_CharT __c, size_type __pos=npo) const noexcept
- size_type `find_last_of` (const `__versa_string` & __str, size_type __pos=npo) const noexcept
- size_type `find_last_of` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `find_last_of` (const _CharT * __s, size_type __pos=npo) const
- const_reference `front` () const noexcept
- reference `front` () noexcept
- allocator_type `get_allocator` () const noexcept
- iterator `insert` (const_iterator __p, _CharT __c)
- template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>>
iterator `insert` (const_iterator __p, _InputIterator __beg, _InputIterator __end)
- iterator `insert` (const_iterator __p, size_type __n, _CharT __c)
- iterator `insert` (const_iterator __p, std::initializer_list<_CharT> __l)
- `__versa_string` & `insert` (size_type __pos, const _CharT * __s)
- `__versa_string` & `insert` (size_type __pos, const _CharT * __s, size_type __n)
- `__versa_string` & `insert` (size_type __pos, size_type __n, _CharT __c)
- `__versa_string` & `insert` (size_type __pos1, const `__versa_string` & __str)
- `__versa_string` & `insert` (size_type __pos1, const `__versa_string` & __str, size_type __pos2, size_type __n)
- size_type `length` () const noexcept
- size_type `max_size` () const noexcept
- `__versa_string` & `operator+=` (_CharT __c)
- `__versa_string` & `operator+=` (const `__versa_string` & __str)
- `__versa_string` & `operator+=` (const _CharT * __s)
- `__versa_string` & `operator+=` (std::initializer_list<_CharT> __l)
- `__versa_string` & `operator=` (_CharT __c) noexcept
- `__versa_string` & `operator=` (const `__versa_string` & __str)
- `__versa_string` & `operator=` (const _CharT * __s)
- `__versa_string` & `operator=` (std::initializer_list<_CharT> __l)
- const_reference `operator[]` (size_type __pos) const noexcept
- reference `operator[]` (size_type __pos) noexcept
- void `pop_back` ()

- void `push_back` (`_CharT __c`)
- `const_reverse_iterator rbegin` () const noexcept
- `reverse_iterator rbegin` () noexcept
- `const_reverse_iterator rend` () const noexcept
- `reverse_iterator rend` () noexcept
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, `_CharT * __k1`, `_CharT * __k2`)
- template<class `_InputIterator` , typename = std::RequireInputIter<`_InputIterator`>>
`__versa_string & replace` (const_iterator __i1, const_iterator __i2, `_InputIterator __k1`, `_InputIterator __k2`)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, const `__versa_string & __str`)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, const `_CharT * __k1`, const `_CharT * __k2`)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, const `_CharT * __s`)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, const `_CharT * __s`, size_type __n)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, const_iterator __k1, const_iterator __k2)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, iterator __k1, iterator __k2)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, size_type __n, `_CharT __c`)
- `__versa_string & replace` (const_iterator __i1, const_iterator __i2, `std::initializer_list<_CharT> __l`)
- `__versa_string & replace` (size_type __pos, size_type __n, const `__versa_string & __str`)
- `__versa_string & replace` (size_type __pos, size_type __n1, const `_CharT * __s`)
- `__versa_string & replace` (size_type __pos, size_type __n1, const `_CharT * __s`, size_type __n2)
- `__versa_string & replace` (size_type __pos, size_type __n1, size_type __n2, `_CharT __c`)
- `__versa_string & replace` (size_type __pos1, size_type __n1, const `__versa_string & __str`, size_type __pos2, size_type __n2)
- void `reserve` (size_type __res_arg=0)
- void `resize` (size_type __n)
- void `resize` (size_type __n, `_CharT __c`)
- size_type `rfind` (`_CharT __c`, size_type __pos=`npos`) const noexcept
- size_type `rfind` (const `__versa_string & __str`, size_type __pos=`npos`) const noexcept
- size_type `rfind` (const `_CharT * __s`, size_type __pos, size_type __n) const
- size_type `rfind` (const `_CharT * __s`, size_type __pos=`npos`) const
- void `shrink_to_fit` () noexcept
- size_type `size` () const noexcept
- `__versa_string substr` (size_type __pos=0, size_type __n=`npos`) const
- void `swap` (`__versa_string & __s`) noexcept

Static Public Attributes

- static const size_type `npos`

6.62.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base>
class __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >
```

Template class `__versa_string`.

Data structure managing sequences of characters and character-like objects.

6.62.2 Constructor & Destructor Documentation

`__versa_string`() [1/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    const _Alloc & __a = _Alloc() ) [inline], [explicit], [noexcept]
```

Construct an empty string using allocator `a`.

`__versa_string()` [2/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Construct string with copy of value of `__str`.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

`__versa_string()` [3/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    __versa_string< _CharT, _Traits, _Alloc, _Base > && __str ) [inline], [noexcept]
```

String move constructor.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

The newly-constructed string contains the exact contents of `__str`. The contents of `__str` are a valid, but unspecified string.

`__versa_string()` [4/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    std::initializer_list< _CharT > __l,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string from an initializer list.

Parameters

<code>__l</code>	<code>std::initializer_list</code> of characters.
<code>__a</code>	Allocator to use (default is default allocator).

`__versa_string()` [5/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos,
    size_type __n = npos ) [inline]
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy (default remainder).

__versa_string() [6/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos,
    size_type __n,
    const _Alloc & __a ) [inline]
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use.

__versa_string() [7/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    const _CharT * __s,
    size_type __n,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string initialized by a character array.

Parameters

<code>__s</code>	Source character array.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use (default is default allocator).

NB: `__s` must have at least `__n` characters, `'\0'` has no special meaning.

__versa_string() [8/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
```

```
const _CharT * __s,
const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a C string.

Parameters

<code>__s</code>	Source C string.
<code>__a</code>	Allocator to use (default is default allocator).

`__versa_string()` [9/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    size_type __n,
    _CharT __c,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as multiple characters.

Parameters

<code>__n</code>	Number of characters.
<code>__c</code>	Character to use.
<code>__a</code>	Allocator to use (default is default allocator).

`__versa_string()` [10/10]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
template<class _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string (
    _InputIterator __beg,
    _InputIterator __end,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a range.

Parameters

<code>__beg</code>	Start of range.
<code>__end</code>	End of range.
<code>__a</code>	Allocator to use (default is default allocator).

`~__versa_string()`

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
```

```
typename > class _Base>
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::~__versa_string ( ) [inline], [noexcept]
Destroy the string instance.
```

6.62.3 Member Function Documentation

append() [1/7]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
template<class _InputIterator , typename = std::RequireInputIter<_InputIterator>>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Append a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Appends characters in the range [first,last) to this string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

append() [2/7]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append\(\)](#), [std::getline\(\)](#), [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator+=\(\(\)\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator+=\(\(\)\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator+=\(\(\)\)](#).

append() [3/7]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
```

```

        size_type __pos,
        size_type __n ) [inline]

```

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of <code>str</code> to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

`append()` [4/7]

```

template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    const _CharT * __s ) [inline]

```

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

`append()` [5/7]

```

template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    const _CharT * __s,
    size_type __n ) [inline]

```

Append a C substring.

Parameters

<code>__s</code>	The C string to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

append() [6/7]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    size_type __n,
    _CharT __c ) [inline]
```

Append multiple characters.

Parameters

<code>↵ _n</code>	The number of characters to append.
<code>↵ _c</code>	The character to use.

Returns

Reference to this string.

Appends n copies of c to this string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

append() [7/7]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
    std::initializer_list< _CharT > __l ) [inline]
```

Append an initializer_list of characters.

Parameters

<code>↵ ↵ ↵ ↵ l</code>	The initializer_list of characters to append.
--	---

Returns

Reference to this string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append\(\)](#).

assign() [1/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (
    __versa_string< _CharT, _Traits, _Alloc, _Base > && __str ) [inline], [noexcept]
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::swap\(\)](#).

assign() [2/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
template<class _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Set value to a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Sets value of string to characters in the range [first,last).

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

assign() [3/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _A](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator=\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base](#) and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator=\(\)](#).

assign() [4/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos,
    size_type __n ) [inline]
```

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

assign() [5/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (
    const _CharT * __s ) [inline]
```

Set value to contents of a C string.

Parameters

<code>__s</code>	The C string to use.
------------------	----------------------

Returns

Reference to this string.

This function sets the value of this string to the value of `__s`. The data is copied, so there is no dependence on `__s` once the function returns.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

assign() [6/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign (
    const _CharT * __s,
    size_type __n ) [inline]
```

Set value to a C substring.

Parameters

<code>__s</code>	The C string to use.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

This function sets the value of this string to the first `__n` characters of `__s`. If `__n` is larger than the number of available characters in `__s`, the remainder of `__s` is used.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

assign() [7/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign (
    size_type __n,
    _CharT __c ) [inline]
```

Set value to multiple characters.

Parameters

<code>__n</code>	Length of the resulting string.
<code>__c</code>	The character to use.

Returns

Reference to this string.

This function sets the value of this string to `__n` copies of character `__c`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

assign() [8/8]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign (
    std::initializer_list<_CharT > __l ) [inline]
```

Set value to an `initializer_list` of characters.

Parameters

\leftrightarrow	The initializer_list of characters to assign.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
$/$	

Returns

Reference to this string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#).

at() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
reference \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base >::at (
    size_type __n ) [inline]
```

Provides access to the data contained in the string.

Parameters

$_ \leftrightarrow$	The index of the character to access.
$_n$	

Returns

Read/write reference to the character.

Exceptions

std::out_of_range	If $_n$ is an invalid index.
-----------------------------------	-------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails. Success results in unsharing the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size\(\)](#).

at() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_reference \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base >::at (
    size_type __n ) const [inline]
```

Provides access to the data contained in the string.

Parameters

$_ \leftrightarrow$	The index of the character to access.
$_n$	

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

back() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::back ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reference to the data at the last element of the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator\[\]\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

back() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::back ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the last element of the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator\[\]\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

begin() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::begin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first character in the string.

begin() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first character in the string. Unshares the string.

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::crend\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rend\(\)](#).

c_str()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const _CharT * __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::c_str ( ) const [inline],
[noexcept]
```

Return const pointer to null-terminated contents.

This is a handle to internal data. Do not modify or dire things may happen.

capacity()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::capacity ( ) const [inline],
[noexcept]
```

Returns the total number of characters that the string can hold before needing to allocate more memory.

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::push_back\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _T](#)

cbegin()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::cbegin ( ) const
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first character in the string.

cend()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::cend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last character in the string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

clear()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::clear ( ) [inline], [noexcept]
```

Erases the string, making it empty.

compare() [1/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) const [inline]
```

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data\(\)](#), [std::min\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _T](#)

Referenced by [__gnu_cxx::operator<\(\)](#), [__gnu_cxx::operator<\(\)](#), [__gnu_cxx::operator<\(\)](#), [__gnu_cxx::operator<=\(\)](#), [__gnu_cxx::operator<=\(\)](#), [__gnu_cxx::operator<=\(\)](#), [__gnu_cxx::operator==\(\)](#), [__gnu_cxx::operator==\(\)](#), [__gnu_cxx::operator==\(\)](#), and [__gnu_cxx::operator==\(\)](#).

[__gnu_cxx::operator>\(\)](#), [__gnu_cxx::operator>\(\)](#), [__gnu_cxx::operator>\(\)](#), [__gnu_cxx::operator>=\(\)](#), [__gnu_cxx::operator>=\(\)](#), and [__gnu_cxx::operator>=\(\)](#).

`compare()` [2/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    const _CharT * __s ) const
```

Compare to a C string.

Parameters

<code>__s</code>	C string to compare against.
------------------	------------------------------

Returns

Integer < 0 , 0 , or > 0 .

Returns an integer < 0 if this string is ordered before `__s`, 0 if their values are equivalent, or > 0 if this string is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and the length of a string constructed from `__s`. The function then compares the two strings by calling `traits::compare(data(),s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [std::min\(\)](#).

`compare()` [3/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    size_type __pos,
    size_type __n,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) const
```

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(),str.data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::data\(\)](#), [std::min\(\)](#), and [__gnu_cxx::__versa_string<_CharT,](#)

compare() [4/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    size_type __pos,
    size_type __n1,
    const _CharT * __s ) const
```

Compare substring to a C string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	C string to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__s`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and the length of a string constructed from `__s`. The function then compares the two string by calling `traits::compare(substring.data(),s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [std::min\(\)](#).

compare() [5/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    size_type __pos,
    size_type __n1,
    const _CharT * __s,
    size_type __n2 ) const
```

Compare substring against a character array.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	character array to compare against.
<code>__n2</code>	Number of characters of s.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos`. Form a string from the first `__n2` characters of `__s`. Returns an integer < 0 if this substring is ordered before the string from `__s`, 0 if their values are equivalent, or > 0 if this substring is ordered after the string from `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__n2`. The function then compares the two strings by calling `traits::compare(substring.data(),__s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is

ordered first.

NB: `__s` must have at least `n2` characters, `l0` has no special meaning.

References [std::min\(\)](#).

compare() [6/6]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (
    size_type __pos1,
    size_type __n1,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos2,
    size_type __n2 ) const
```

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of str.
<code>__n2</code>	Number of characters in substring of str.

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(),str.substr(pos2,n2).data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::data\(\)](#), and [std::min\(\)](#).

copy()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::copy (
    _CharT * __s,
    size_type __n,
    size_type __pos = 0 ) const
```

Copy substring into C string.

Parameters

<code>__s</code>	C string to copy value into.
<code>__n</code>	Number of characters to copy.
<code>__pos</code>	Index of first character to copy.

Returns

Number of characters actually copied

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
--------------------------------	-----------------------------------

Copies up to `__n` characters starting at `__pos` into the C string `s`. If `__pos` is greater than `size()`, `out_of_range` is thrown.

crbegin()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::crbegin ( )
const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::end\(\)](#).

crend()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::crend ( )
const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::begin\(\)](#).

data()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const _CharT * __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data ( ) const [inline],
[noexcept]
```

Return const pointer to contents.

This is a handle to internal data. Do not modify or dire things may happen.

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of\(\)](#).

empty()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
bool __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::empty ( ) const [inline],
[noexcept]
```

Returns true if the string is empty. Equivalent to `*this == ""`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

end() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_iterator \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base>::end \( \) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last character in the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

end() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base>::end \( \) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last character in the string. Unshares the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::crbegin\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _A](#) and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rbegin\(\)](#).

erase() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base>::erase \(
    const_iterator __first,
    const_iterator __last ) [inline]
```

Remove a range of characters.

Parameters

<code><i>__first</i></code>	Iterator referencing the first character to remove.
<code><i>__last</i></code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range [first,last) from this string. The value of the string doesn't change if an error is thrown.

erase() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator \_\_gnu\_cxx::\_\_versa\_string<\_CharT, \_Traits, \_Alloc, \_Base>::erase \(
    const_iterator __position ) [inline]
```

Remove one character.

Parameters

<code><i>__position</i></code>	Iterator referencing the character to remove.
--------------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

erase() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::erase (
    size_type __pos = 0,
    size_type __n = npos ) [inline]
```

Remove characters.

Parameters

<code>__pos</code>	Index of first character to remove (default 0).
<code>__n</code>	Number of characters to remove (default remainder).

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.
--------------------------------	---

Removes `__n` characters from this string starting at `__pos`. The length of the string is reduced by `__n`. If there are $< __n$ characters to remove, the remainder of the string is truncated. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Referenced by [std::getline\(\)](#).

find() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::find (
    _CharT __c,
    size_type __pos = 0 ) const [noexcept]
```

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type \_\_gnu\_cxx::\_\_versa\_string< _CharT, _Traits, _Alloc, _Base >::find (
    const \_\_versa\_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#) and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

find() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
\_\_versa\_string< _CharT, _Traits, _Alloc, _Base >::size_type \_\_gnu\_cxx::\_\_versa\_string< _CharT, ↵
_Traits, _Alloc, _Base >::find (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const
```

Find position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>__s</code> to search for.

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of\(\)](#) and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of\(\)](#).

find() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type \_\_gnu\_cxx::\_\_versa\_string< _CharT, _Traits, _Alloc, _Base >::find (
```

```
const _CharT * __s,
size_type __pos = 0 ) const [inline]
```

Find position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find\(\)](#).

find_first_not_of() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::find_first_not_of (
    _CharT __c,
    size_type __pos = 0 ) const [noexcept]
```

Find position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find_first_not_of() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of\(\)](#).

find_first_not_of() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::find_first_not_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const
```

Find position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>s</code> to consider.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find_first_not_of() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of (
    const _CharT * __s,
    size_type __pos = 0 ) const [inline]
```

Find position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of\(\)](#).

find_first_of() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of (
    _CharT __c,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for the character `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `find(c, pos)`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find\(\)](#).

find_first_of() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of\(\)](#).

find_first_of() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of (
    const _CharT * __s,
```

```
size_type __pos,
size_type __n ) const
```

Find position of a character of C substring.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>s</code> to search for.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

`find_first_of()` [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of (
    const _CharT * __s,
    size_type __pos = 0 ) const [inline]
```

Find position of a character of C string.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of\(\)](#).

`find_last_not_of()` [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::find_last_not_of (
    _CharT __c,
    size_type __pos = npos ) const [noexcept]
```

Find last position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find_last_not_of() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_not_of (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_not_of\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_not_of\(\)](#), and [__gnu_cxx::__versa_string< _Cha](#)

find_last_not_of() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::find_last_not_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const
```

Find last position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to consider.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find_last_not_of() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_not_of (
    const _CharT * __s,
    size_type __pos = npos ) const [inline]
```

Find last position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of\(\)](#).

find_last_of() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of (
    _CharT __c,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `rfind(c, pos)`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind\(\)](#).

find_last_of() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of\(\)](#).

find_last_of() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const
```

Find last position of a character of C substring.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

find_last_of() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of (
    const _CharT * __s,
    size_type __pos = npos ) const [inline]
```

Find last position of a character of C string.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of\(\)](#).

front() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const_reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::front ( ) const
[inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator\[\]\(\)](#).

front() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::front ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator\[\]\(\)](#).

get_allocator()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
allocator_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::get_allocator ( )
const [inline], [noexcept]
```

Return copy of allocator used to construct this string.

insert() [1/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    const_iterator __p,
    _CharT __c ) [inline]
```

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

insert() [2/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
template<class _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    const_iterator __p,
    _InputIterator __beg,
    _InputIterator __end ) [inline]
```

Insert a range of characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Returns

Iterator referencing the first inserted char.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Inserts characters in range `[beg,end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

insert() [3/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    const_iterator __p,
    size_type __n,
    _CharT __c ) [inline]
```

Insert multiple characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Iterator referencing the first inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert\(\)](#).

insert() [4/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    const_iterator __p,
    std::initializer_list< _CharT > __l ) [inline]
```

Insert an initializer_list of characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__l</code>	The initializer_list of characters to insert.

Returns

Iterator referencing the first inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert\(\)](#).

insert() [5/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
```

```

typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    size_type __pos,
    const _CharT * __s ) [inline]

```

Insert a C string.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts the first `__n` characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

`insert()` [6/9]

```

template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    size_type __pos,
    const _CharT * __s,
    size_type __n ) [inline]

```

Insert a C substring.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts the first `__n` characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

insert() [7/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    size_type __pos,
    size_type __n,
    _CharT __c ) [inline]
```

Insert multiple characters.

Parameters

<code>__pos</code>	Index in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts `__n` copies of character `__c` starting at index `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos > length()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

insert() [8/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    size_type __pos1,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Insert value of a string.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _](#)

insert() [9/9]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (
    size_type __pos1,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos2,
    size_type __n ) [inline]
```

Insert a substring.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .

Starting at `__pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#).

length()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::length ( ) const [inline],
[noexcept]
```

Returns the number of characters in the string, not including any null-termination.

max_size()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::max_size ( ) const [inline],
[noexcept]
```

Returns the size() of the largest possible string.

Referenced by `std::getline()`.

operator+=() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+= (
    _CharT __c ) [inline]
```

Append a character.

Parameters

<code>__c</code>	The character to append.
------------------	--------------------------

Returns

Reference to this string.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back()`.

operator+=() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+= (
    const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str ) [inline]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`.

operator+=() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+= (
    const _CharT * __s ) [inline]
```

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append\(\)](#).

operator+=() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator+= (
    std::initializer_list< _CharT > __l ) [inline]
```

Append an initializer_list of characters.

Parameters

↵	The initializer_list of characters to be appended.
↵	
↵	
↵	
/	

Returns

Reference to this string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append\(\)](#).

operator=() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator= (
    __versa_string< _CharT, _Traits, _Alloc, _Base > && __str ) [inline], [noexcept]
```

String move assignment operator.

Parameters

__str	Source string.
-------	----------------

The contents of __str are moved into this string (without copying). __str is a valid, but unspecified string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::swap\(\)](#).

operator=() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator= (
    _CharT __c ) [inline]
```

Set value to string of length 1.

Parameters

↵	Source character.
_c	

Assigning to a character makes this string length 1 and `(*this)[0] == __c`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#).

operator=() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator= (
    const __versa_string<_CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Assign the value of *str* to this string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#).

operator=() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator= (
    const _CharT * __s ) [inline]
```

Copy contents of *s* into this string.

Parameters

<code>__s</code>	Source null-terminated string.
------------------	--------------------------------

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#).

operator=() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator= (
    std::initializer_list<_CharT > __l ) [inline]
```

Set value to string constructed from initializer list.

Parameters

<code>__l</code>	<code>std::initializer_list</code> .
------------------	--------------------------------------

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign\(\)](#).

operator[]() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
const_reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator[] (
    size_type __pos ) const [inline], [noexcept]
```

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read-only (constant) reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::back\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::front\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator\[\]\(\)](#).

`operator[]()` [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator[] (
    size_type __pos ) [inline], [noexcept]
```

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read/write reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.) Unshares the string.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

`pop_back()`

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::pop_back ( ) [inline]
```

Remove the last character.

The string must be non-empty.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size\(\)](#).

`push_back()`

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::push_back (
    _CharT __c ) [inline]
```

Append a single character.

Parameters

<code>_↔</code>	Character to append.
<code>_C</code>	

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::capacity\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits,](#)

Referenced by [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::operator+\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::oper](#)

rbegin() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
const_reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rbegin ( )
```

```
const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::end\(\)](#).

rbegin() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rbegin ( ) [inline],
[noexcept]
```

Returns a read/write reverse iterator that points to the last character in the string. Iteration is done in reverse element order. Unshares the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::end\(\)](#).

rend() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
const_reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rend ( )
```

```
const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::begin\(\)](#).

rend() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rend ( ) [inline],
[noexcept]
```

Returns a read/write reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order. Unshares the string.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::begin\(\)](#).

replace() [1/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
```

```
template<class _InputIterator , typename = std::RequireInputIter<_InputIterator>>
```

```
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __il,
```

```
const_iterator __i2,
_InputIterator __k1,
_InputIterator __k2 ) [inline]
```

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, characters in the range `[k1,k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [2/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __i1,
    const_iterator __i2,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, the value of `__str` is inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace()`, and `__gnu_cxx::__versa_string< _CharT, _Traits,`

replace() [3/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __i1,
    const_iterator __i2,
    const _CharT * __s ) [inline]
```

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

replace() [4/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __i1,
    const_iterator __i2,
    const _CharT * __s,
    size_type __n ) [inline]
```

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from <code>s</code> to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, the first *n* characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#).

replace() [5/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __i1,
    const_iterator __i2,
    size_type __n,
    _CharT __c ) [inline]
```

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [6/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    const_iterator __i1,
    const_iterator __i2,
```

```
std::initializer_list<_CharT> __l ) [inline]
```

Replace range of characters with `initializer_list`.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, characters in the range `[k1,k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

`replace()` [7/11]

```
template<typename _CharT, typename _Traits, typename _Alloc, template<typename, typename,
typename> class _Base>
```

```
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace (
    size_type __pos,
    size_type __n,
    const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str ) [inline]
```

Replace characters with value from another string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos,pos+n)` from this string. In place, the value of `__str` is inserted. If `__pos` is

beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#).

replace() [8/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    size_type __pos,
    size_type __n1,
    const _CharT * __s ) [inline]
```

Replace characters with value of a C string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, the characters of `__s` are inserted. If `pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown.

The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace\(\)](#).

replace() [9/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    size_type __pos,
    size_type __n1,
    const _CharT * __s,
    size_type __n2 ) [inline]
```

Replace characters with value of a C substring.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.
<code>__n2</code>	Number of characters from <code>__s</code> to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos1 > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, the first `__n2` characters of `__s` are inserted, or all of `__s` if `__n2` is too large. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [10/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace (
    size_type __pos,
    size_type __n1,
    size_type __n2,
    _CharT __c ) [inline]
```

Replace characters with multiple characters.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__n2</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, `__n2` copies of `__c` are inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [11/11]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string & __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
    size_type __pos1,
    size_type __n1,
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos2,
    size_type __n2 ) [inline]
```

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of str to use.
<code>__n2</code>	Number of characters from str to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > str.size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos1, pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace\(\)](#).

reserve()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::reserve (
    size_type __res_arg = 0 ) [inline]
```

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<code>std::length_error</code>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number

requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

Referenced by [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::operator+\(\)](#), [__gnu_cxx::operator+\(\)](#), and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+\(\)](#).

resize() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize (
    size_type __n ) [inline]
```

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
------------------	---

This function will resize the string to the specified length. If the new size is smaller than the string's current size the string is truncated, otherwise the string is extended and new characters are default-constructed. For basic types such as `char`, this means setting them to 0.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize\(\)](#).

resize() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize (
    size_type __n,
    _CharT __c )
```

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
<code>__c</code>	Character to fill any new elements.

This function will resize the string to the specified number of characters. If the number is smaller than the string's current size the string is truncated, otherwise the string is extended and new elements are set to `__c`.

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize\(\)](#).

rfind() [1/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind (
    _CharT __c,
    size_type __pos = npos ) const [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

rfind() [2/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind (
    const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind\(\)](#).

Referenced by [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of\(\)](#), [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of\(\)](#), and [__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind\(\)](#).

rfind() [3/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, ↵
_Traits, _Alloc, _Base >::rfind (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const
```

Find last position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::min\(\)](#).

rfind() [4/4]

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind (
    const _CharT * __s,
    size_type __pos = npos ) const [inline]
```

Find last position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to start search at (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind\(\)](#).

shrink_to_fit()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::shrink_to_fit ( ) [inline],
[noexcept]
```

A non-binding request to reduce capacity() to size().

References [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::capacity\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#) and [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size\(\)](#).

size()

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size ( ) const [inline],
[noexcept]
```

Returns the number of characters in the string, not including any null-termination.

Referenced by [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::at\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::at\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::back\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::ba](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::cend\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::co](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end\(\)](#), [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end\(\)](#).


```

__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find(), __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of(), __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Bas
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert(), __gnu_cxx::operator+(), __gnu_cxx::operator+(),
__gnu_cxx::operator+(), __gnu_cxx::operator+(), __gnu_cxx::operator+(), __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>:
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[](), __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back(), __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace(), and __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Bas

```

substr()

```

template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
__versa_string __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::substr (
    size_type __pos = 0,
    size_type __n = npos ) const [inline]

```

Get a substring.

Parameters

<code>__pos</code>	Index of first character (default 0).
<code>__n</code>	Number of characters in substring (default remainder).

Returns

The new string.

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
--------------------------------	-----------------------------------

Construct and return a new string using the `__n` characters starting at `__pos`. If the string is too short, use the remainder of the characters. If `__pos` is beyond the end of the string, `out_of_range` is thrown.

swap()

```

template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::swap (
    __versa_string<_CharT, _Traits, _Alloc, _Base> & __s ) [inline], [noexcept]

```

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _A` and `__gnu_cxx::swap()`.

6.62.4 Member Data Documentation

npos

```
template<typename _CharT , typename _Traits , typename _Alloc , template< typename, typename,
typename > class _Base>
const __versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _↔
CharT, _Traits, _Alloc, _Base >::npos [static]
```

Value returned by various member functions when they fail.

The documentation for this class was generated from the following files:

- [vstring.h](#)
- [vstring.tcc](#)

6.63 `__gnu_debug::_After_nth_from<_Iterator>` Class Template Reference

```
#include <safe_sequence.h>
```

Public Member Functions

- `_After_nth_from` (const difference_type &__n, const _Iterator &__base)
- bool `operator()` (const _Iterator &__x) const

6.63.1 Detailed Description

```
template<typename _Iterator>
class __gnu_debug::_After_nth_from<_Iterator>
```

A function object that returns true when the given random access iterator is at least `n` steps away from the given iterator.
The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

6.64 `std::_Base_bitset<_Nw>` Struct Template Reference

Public Types

- typedef unsigned long `_WordT`

Public Member Functions

- constexpr `_Base_bitset` (unsigned long long __val) noexcept
- template<size_t _Nb>
bool `_M_are_all` () const noexcept
- void `_M_do_and` (const [_Base_bitset](#)<_Nw> &__x) noexcept
- size_t `_M_do_count` () const noexcept
- size_t `_M_do_find_first` (size_t) const noexcept
- size_t `_M_do_find_next` (size_t, size_t) const noexcept
- void `_M_do_flip` () noexcept
- void `_M_do_left_shift` (size_t __shift) noexcept
- void `_M_do_or` (const [_Base_bitset](#)<_Nw> &__x) noexcept
- void `_M_do_reset` () noexcept
- void `_M_do_right_shift` (size_t __shift) noexcept
- void `_M_do_set` () noexcept
- unsigned long long `_M_do_to_ullong` () const
- unsigned long `_M_do_to_ulong` () const

- void **_M_do_xor** (const [_Base_bitset](#)< _Nw > &__x) noexcept
- const _WordT * **_M_getdata** () const noexcept
- constexpr _WordT **_M_getword** (size_t __pos) const noexcept
- _WordT & **_M_getword** (size_t __pos) noexcept
- constexpr _WordT **_M_hiword** () const noexcept
- _WordT & **_M_hiword** () noexcept
- bool **_M_is_any** () const noexcept
- bool **_M_is_equal** (const [_Base_bitset](#)< _Nw > &__x) const noexcept

Static Public Member Functions

- static constexpr _WordT **_S_maskbit** (size_t __pos) noexcept
- static constexpr size_t **_S_whichbit** (size_t __pos) noexcept
- static constexpr size_t **_S_whichbyte** (size_t __pos) noexcept
- static constexpr size_t **_S_whichword** (size_t __pos) noexcept

Public Attributes

- _WordT [_M_w](#) [_Nw]

6.64.1 Detailed Description

template<size_t _Nw>
struct std::_Base_bitset< _Nw >

Base class, general case. It is a class invariant that _Nw will be nonnegative.
 See documentation for bitset.

6.64.2 Member Data Documentation

_M_w

template<size_t _Nw>
 _WordT [std::_Base_bitset](#)< _Nw >::_M_w[_Nw]

0 is the least significant word.

The documentation for this struct was generated from the following file:

- [bitset](#)

6.65 std::_Base_bitset< 0 > Struct Reference

Public Types

- typedef unsigned long **_WordT**

Public Member Functions

- constexpr **_Base_bitset** (unsigned long long) noexcept
- template<size_t _Nb>
 bool **_M_are_all** () const noexcept
- void **_M_do_and** (const [_Base_bitset](#)< 0 > &) noexcept
- size_t **_M_do_count** () const noexcept
- size_t **_M_do_find_first** (size_t) const noexcept
- size_t **_M_do_find_next** (size_t, size_t) const noexcept
- void **_M_do_flip** () noexcept

- `void _M_do_left_shift (size_t) noexcept`
- `void _M_do_or (const _Base_bitset< 0 > &) noexcept`
- `void _M_do_reset () noexcept`
- `void _M_do_right_shift (size_t) noexcept`
- `void _M_do_set () noexcept`
- `unsigned long long _M_do_to_ullong () const noexcept`
- `unsigned long _M_do_to_ulong () const noexcept`
- `void _M_do_xor (const _Base_bitset< 0 > &) noexcept`
- `constexpr _WordT _M_getword (size_t) const noexcept`
- `_WordT & _M_getword (size_t) noexcept`
- `constexpr _WordT _M_hiword () const noexcept`
- `bool _M_is_any () const noexcept`
- `bool _M_is_equal (const _Base_bitset< 0 > &) const noexcept`

Static Public Member Functions

- `static constexpr _WordT _S_maskbit (size_t __pos) noexcept`
- `static constexpr size_t _S_whichbit (size_t __pos) noexcept`
- `static constexpr size_t _S_whichbyte (size_t __pos) noexcept`
- `static constexpr size_t _S_whichword (size_t __pos) noexcept`

6.65.1 Detailed Description

Base class, specialization for no storage (zero-length bitset).

See documentation for `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

6.66 `std::_Base_bitset< 1 >` Struct Reference

Public Types

- `typedef unsigned long _WordT`

Public Member Functions

- `constexpr _Base_bitset (unsigned long long __val) noexcept`
- `template<size_t _Nb>`
`bool _M_are_all () const noexcept`
- `void _M_do_and (const _Base_bitset< 1 > &__x) noexcept`
- `size_t _M_do_count () const noexcept`
- `size_t _M_do_find_first (size_t __not_found) const noexcept`
- `size_t _M_do_find_next (size_t __prev, size_t __not_found) const noexcept`
- `void _M_do_flip () noexcept`
- `void _M_do_left_shift (size_t __shift) noexcept`
- `void _M_do_or (const _Base_bitset< 1 > &__x) noexcept`
- `void _M_do_reset () noexcept`
- `void _M_do_right_shift (size_t __shift) noexcept`
- `void _M_do_set () noexcept`
- `unsigned long long _M_do_to_ullong () const noexcept`
- `unsigned long _M_do_to_ulong () const noexcept`
- `void _M_do_xor (const _Base_bitset< 1 > &__x) noexcept`

- `const _WordT * _M_getdata () const noexcept`
- `constexpr _WordT _M_getword (size_t) const noexcept`
- `_WordT & _M_getword (size_t) noexcept`
- `constexpr _WordT _M_hiword () const noexcept`
- `_WordT & _M_hiword () noexcept`
- `bool _M_is_any () const noexcept`
- `bool _M_is_equal (const _Base_bitset< 1 > &__x) const noexcept`

Static Public Member Functions

- `static constexpr _WordT _S_maskbit (size_t __pos) noexcept`
- `static constexpr size_t _S_whichbit (size_t __pos) noexcept`
- `static constexpr size_t _S_whichbyte (size_t __pos) noexcept`
- `static constexpr size_t _S_whichword (size_t __pos) noexcept`

Public Attributes

- `_WordT _M_w`

6.66.1 Detailed Description

Base class, specialization for a single word.

See documentation for `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

6.67 `__gnu_debug::_BeforeBeginHelper< _Sequence >` Struct Template Reference

```
#include <safe_iterator.h>
```

Static Public Member Functions

- `template<typename _Iterator, typename _Category >
static bool _S_Is (const _Safe_iterator< _Iterator, _Sequence, _Category > &)`
- `template<typename _Iterator, typename _Category >
static bool _S_Is_Beginnest (const _Safe_iterator< _Iterator, _Sequence, _Category > &__it)`

6.67.1 Detailed Description

```
template<typename _Sequence>
```

```
struct __gnu_debug::_BeforeBeginHelper< _Sequence >
```

Helper struct to deal with sequence offering a `before_begin` iterator.

The documentation for this struct was generated from the following file:

- [safe_iterator.h](#)

6.68 `std::_Bind< _Signature >` Class Template Reference

6.68.1 Detailed Description

```
template<typename _Signature>
```

```
class std::_Bind< _Signature >
```

Type of the function object returned from `bind()`.

The documentation for this class was generated from the following file:

- [functional](#)

6.69 `std::_Bind_result<_Result, _Signature >` Class Template Reference

6.69.1 Detailed Description

```
template<typename _Result, typename _Signature>
class std::_Bind_result<_Result, _Signature >
```

Type of the function object returned from `bind<R>()`.

The documentation for this class was generated from the following file:

- [functional](#)

6.70 `__gnu_cxx::__detail::_Bitmap_counter<_Tp >` Class Template Reference

```
#include <bitmap_allocator.h>
```

Public Member Functions

- `_Bitmap_counter` ([_BPVector](#) &Rvbp, long __index=-1)
- `pointer _M_base` () const throw ()
- `bool _M_finished` () const throw ()
- `std::size_t * _M_get` () const throw ()
- `_Index_type _M_offset` () const throw ()
- `void _M_reset` (long __index=-1) throw ()
- `void _M_set_internal_bitmap` (std::size_t * __new_internal_marker) throw ()
- `_Index_type _M_where` () const throw ()
- `_Bitmap_counter & operator++` () throw ()

6.70.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__detail::_Bitmap_counter<_Tp >
```

The bitmap counter which acts as the bitmap manipulator, and manages the bit-manipulation functions and the searching and identification functions on the bit-map.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

6.71 `std::__detail::_BracketMatcher<_TraitsT, __icase, __collate >` Struct Template Reference

```
#include <regex_compiler.h>
```

Public Types

- `typedef _TraitsT::char_class_type _CharClassT`
- `typedef _TransT::_CharT _CharT`
- `typedef _TraitsT::string_type _StringT`
- `typedef _TransT::_StrTransT _StrTransT`
- `typedef _RegexTranslator<_TraitsT, __icase, __collate > _TransT`

Public Member Functions

- **_BracketMatcher** (bool __is_non_matching, const _TraitsT &__traits)
- void **_M_add_char** (_CharT __c)
- void **_M_add_character_class** (const _StringT &__s, bool __neg)
- _StringT **_M_add_collate_element** (const _StringT &__s)
- void **_M_add_equivalence_class** (const _StringT &__s)
- void **_M_make_range** (_CharT __l, _CharT __r)
- void **_M_ready** ()
- bool **operator()** (_CharT __ch) const

6.71.1 Detailed Description

```
template<typename _TraitsT, bool __icase, bool __collate>
struct std::__detail::_BracketMatcher< _TraitsT, __icase, __collate >
```

Matches a character range (bracket expression)

The documentation for this struct was generated from the following files:

- [regex_compiler.h](#)
- [regex_compiler.tcc](#)

6.72 __gnu_cxx::_Caster<_ToType> Struct Template Reference

```
#include <cast.h>
```

Public Types

- typedef _ToType::element_type * **type**

6.72.1 Detailed Description

```
template<typename _ToType>
struct __gnu_cxx::_Caster< _ToType >
```

These functions are here to allow containers to support non standard pointer types. For normal pointers, these resolve to the use of the standard cast operation. For other types the functions will perform the appropriate cast to/from the custom pointer class so long as that class meets the following conditions: 1) has a typedef element_type which names the type it points to. 2) has a get() const method which returns element_type*. 3) has a constructor which can take one element_type* argument. This type supports the semantics of the pointer cast operators (below.)

The documentation for this struct was generated from the following file:

- [cast.h](#)

6.73 __gnu_cxx::_Char_traits<_CharT> Struct Template Reference

```
#include <char_traits.h>
```

Public Types

- typedef unsigned long **int_type**
- typedef [std::streamoff](#) **off_type**
- typedef [std::streampos](#) **pos_type**
- typedef [std::mbstate_t](#) **state_type**

6.73.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::_Char_types< _CharT >
```

Mapping from character type to associated types.

Note

This is an implementation class for the generic version of `char_traits`. It defines `int_type`, `off_type`, `pos_type`, and `state_type`. By default these are unsigned long, streamoff, streampos, and mbstate_t. Users who need a different set of types, but who don't need to change the definitions of any function defined in `char_traits`, can specialize `__gnu_cxx::_Char_types` while leaving `__gnu_cxx::char_traits` alone.

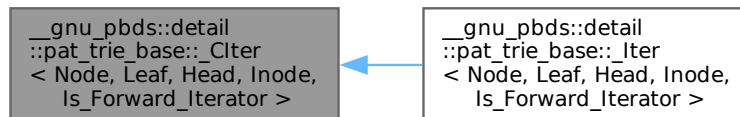
The documentation for this struct was generated from the following file:

- [char_traits.h](#)

6.74 `__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >` Class Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`:



Public Types

- typedef allocator_type **_Alloc**
- typedef Node::allocator_type **allocator_type**
- typedef type_traits::const_pointer **const_pointer**
- typedef type_traits::const_reference **const_reference**
- typedef allocator_type::difference_type **difference_type**
- typedef [rebind_traits](#)< _Alloc, Head >::pointer **head_pointer**
- typedef Inode::iterator **inode_iterator**
- typedef [rebind_traits](#)< _Alloc, Inode >::pointer **inode_pointer**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef [rebind_traits](#)< _Alloc, Leaf >::const_pointer **leaf_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Leaf >::pointer **leaf_pointer**
- typedef [rebind_traits](#)< _Alloc, Node >::pointer **node_pointer**
- typedef type_traits::pointer **pointer**
- typedef type_traits::reference **reference**
- typedef Node::type_traits **type_traits**
- typedef type_traits::value_type **value_type**

Public Member Functions

- `_Clter` (const `_Clter`< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other)
- `_Clter` (node_pointer p_nd=0)
- `bool operator!=` (const `_Clter` &other) const
- `bool operator!=` (const `_Clter`< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other) const
- `const_reference operator*` () const
- `_Clter & operator++` ()
- `_Clter operator++` (int)
- `_Clter & operator--` ()
- `_Clter operator--` (int)
- `const_pointer operator->` () const
- `_Clter & operator=` (const `_Clter` &other)
- `_Clter & operator=` (const `_Clter`< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other)
- `bool operator==` (const `_Clter` &other) const
- `bool operator==` (const `_Clter`< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other) const

Public Attributes

- node_pointer `m_p_nd`

Protected Member Functions

- void `dec` (false_type)
- void `dec` (true_type)
- void `inc` (false_type)
- void `inc` (true_type)

Static Protected Member Functions

- static node_pointer `get_larger_sibling` (node_pointer p_nd)
- static node_pointer `get_smaller_sibling` (node_pointer p_nd)
- static leaf_pointer `leftmost_descendant` (node_pointer p_nd)
- static leaf_pointer `rightmost_descendant` (node_pointer p_nd)

6.74.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, bool Is_Forward_Iterator>
class __gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >
```

Const iterator.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

6.75 std::__detail::Compiler< _TraitsT > Class Template Reference

```
#include <regex_compiler.h>
```

Public Types

- typedef `_TraitsT::char_type` `_CharT`
- typedef `regex_constants::syntax_option_type` `_FlagT`
- typedef `_NFA< _TraitsT >` `_RegexT`

Public Member Functions

- **_Compiler** (const _CharT *__b, const _CharT *__e, const typename _TraitsT::locale_type &__traits, _FlagT __flags)
- **shared_ptr**< const _RegexT > **_M_get_nfa** () noexcept

6.75.1 Detailed Description

```
template<typename _TraitsT>
class std::__detail::_Compiler<_TraitsT >
```

Builds an NFA from an input iterator range.

The _TraitsT type should fulfill requirements [28.3].

The documentation for this class was generated from the following files:

- [regex_compiler.h](#)
- [regex_compiler.tcc](#)

6.76 std::__parallel::_CRandNumber<_MustBeInt > Struct Template Reference

```
#include <algo.h>
```

Public Member Functions

- **operator()** (int __limit)

6.76.1 Detailed Description

```
template<typename _MustBeInt = int>
struct std::__parallel::_CRandNumber<_MustBeInt >
```

Functor wrapper for std::rand().

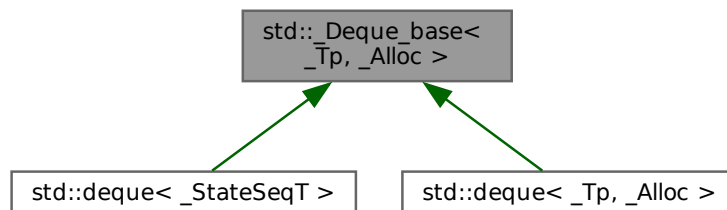
The documentation for this struct was generated from the following file:

- [algo.h](#)

6.77 std::_Deque_base<_Tp, _Alloc > Class Template Reference

```
#include <stl_deque.h>
```

Inheritance diagram for std::_Deque_base<_Tp, _Alloc >:



Protected Types

- enum { **_S_initial_map_size** }
- typedef [__gnu_cxx::__alloc_traits](#)< _Tp_alloc_type > **_Alloc_traits**
- typedef [__gnu_cxx::__alloc_traits](#)< _Map_alloc_type > **_Map_alloc_traits**
- typedef _Alloc_traits::template rebind< _Ptr >::other **_Map_alloc_type**
- typedef iterator:: _Map_pointer **_Map_pointer**
- typedef _Alloc_traits::pointer **_Ptr**
- typedef _Alloc_traits::const_pointer **_Ptr_const**
- typedef [__gnu_cxx::__alloc_traits](#)< _Alloc >::template rebind< _Tp >::other **_Tp_alloc_type**
- typedef _Alloc **allocator_type**
- typedef [_Deque_iterator](#)< _Tp, const _Tp &, _Ptr_const > **const_iterator**
- typedef [_Deque_iterator](#)< _Tp, _Tp &, _Ptr > **iterator**

Protected Member Functions

- [_Deque_base](#) ([_Deque_base](#) &&__x)
- [_Deque_base](#) ([_Deque_base](#) &&__x, const allocator_type &__a)
- [_Deque_base](#) ([_Deque_base](#) &&__x, const allocator_type &__a, size_t __n)
- [_Deque_base](#) (const allocator_type &__a)
- [_Deque_base](#) (const allocator_type &__a, size_t __num_elements)
- [_Deque_base](#) (size_t __num_elements)
- [_Map_pointer](#) **_M_allocate_map** (size_t __n)
- [_Ptr](#) **_M_allocate_node** ()
- void **_M_create_nodes** ([_Map_pointer](#) __nstart, [_Map_pointer](#) __nfinish)
- void **_M_deallocate_map** ([_Map_pointer](#) __p, size_t __n) noexcept
- void **_M_deallocate_node** ([_Ptr](#) __p) noexcept
- void **_M_destroy_nodes** ([_Map_pointer](#) __nstart, [_Map_pointer](#) __nfinish) noexcept
- [_Map_alloc_type](#) **_M_get_map_allocator** () const noexcept
- const _Tp_alloc_type & **_M_get_Tp_allocator** () const noexcept
- [_Tp_alloc_type](#) & **_M_get_Tp_allocator** () noexcept
- void [_M_initialize_map](#) (size_t)
- allocator_type **get_allocator** () const noexcept

Protected Attributes

- [_Deque_impl](#) [_M_impl](#)

6.77.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
class std::_Deque_base< _Tp, _Alloc >
```

Deque base class. This class provides the unified face for deque's allocation. This class's constructor and destructor allocate and deallocate (but do not initialize) storage. This makes exception safety easier.

Nothing in this class ever constructs or destroys an actual Tp element. (Deque handles that itself.) Only/All memory management is performed here.

6.77.2 Member Function Documentation

_M_initialize_map()

```
template<typename _Tp , typename _Alloc >  
void std::_Deque_base<_Tp, _Alloc>::_M_initialize_map (  
    size_t __num_elements ) [protected]
```

Layout storage.

Parameters

<code>__num_elements</code>	The count of T's for which to allocate space at first.
-----------------------------	--

Returns

Nothing.

The initial underlying memory layout is a bit complicated...

References [std::max\(\)](#).

The documentation for this class was generated from the following file:

- [stl_deque.h](#)

6.78 `std::_Deque_iterator<_Tp, _Ref, _Ptr>` Struct Template Reference

```
#include <stl_deque.h>
```

Public Types

- typedef `__ptr_rebind<_Ptr, _Tp> _Elt_pointer`
- typedef `__ptr_rebind<_Ptr, _Elt_pointer> _Map_pointer`
- typedef `_Deque_iterator _Self`
- typedef `__iter<const _Tp> const_iterator`
- typedef `ptrdiff_t difference_type`
- typedef `__iter<_Tp> iterator`
- typedef `std::random_access_iterator_tag iterator_category`
- typedef `_Ptr pointer`
- typedef `_Ref reference`
- typedef `size_t size_type`
- typedef `_Tp value_type`

Public Member Functions

- `_Deque_iterator` (`_Elt_pointer __x, _Map_pointer __y`) noexcept
- `_Deque_iterator` (`const _Deque_iterator &__x`) noexcept
- `template<typename _Iter, typename = _Require<is_same<_Self, const_iterator>, is_same<_Iter, iterator>>>`
`_Deque_iterator` (`const _Iter &__x`) noexcept
- `iterator _M_const_cast` () const noexcept
- `void _M_set_node` (`_Map_pointer __new_node`) noexcept
- reference `operator*` () const noexcept
- `_Self & operator++` () noexcept
- `_Self operator++` (int) noexcept
- `_Self & operator+=` (`difference_type __n`) noexcept
- `_Self & operator--` () noexcept
- `_Self operator--` (int) noexcept
- `_Self & operator-=` (`difference_type __n`) noexcept
- pointer `operator->` () const noexcept
- `_Deque_iterator & operator=` (`const _Deque_iterator &`)=default
- reference `operator[]` (`difference_type __n`) const noexcept

Static Public Member Functions

- static `size_t _S_buffer_size` () noexcept

Public Attributes

- `_Elt_pointer _M_cur`
- `_Elt_pointer _M_first`
- `_Elt_pointer _M_last`
- `_Map_pointer _M_node`

Friends

- `template<typename _RefR, typename _PtrR >`
`bool operator!= (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator!= (const _Self &__x, const _Self &__y) noexcept`
- `_Self operator+ (const _Self &__x, difference_type __n) noexcept`
- `_Self operator+ (difference_type __n, const _Self &__x) noexcept`
- `template<typename _RefR, typename _PtrR >`
`difference_type operator- (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `difference_type operator- (const _Self &__x, const _Self &__y) noexcept`
- `_Self operator- (const _Self &__x, difference_type __n) noexcept`
- `template<typename _RefR, typename _PtrR >`
`bool operator< (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator< (const _Self &__x, const _Self &__y) noexcept`
- `template<typename _RefR, typename _PtrR >`
`bool operator<= (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator<= (const _Self &__x, const _Self &__y) noexcept`
- `template<typename _RefR, typename _PtrR >`
`bool operator== (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator== (const _Self &__x, const _Self &__y) noexcept`
- `template<typename _RefR, typename _PtrR >`
`bool operator> (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator> (const _Self &__x, const _Self &__y) noexcept`
- `template<typename _RefR, typename _PtrR >`
`bool operator>= (const _Self &__x, const Deque_iterator<_Tp, _RefR, _PtrR> &__y) noexcept`
- `bool operator>= (const _Self &__x, const _Self &__y) noexcept`

6.78.1 Detailed Description

`template<typename _Tp, typename _Ref, typename _Ptr>`

`struct std::Deque_iterator<_Tp, _Ref, _Ptr>`

A deque::iterator.

Quite a bit of intelligence here. Much of the functionality of deque is actually passed off to this class. A deque holds two of these internally, marking its valid range. Access to elements is done as offsets of either of those two, relying on operator overloading in this class.

All the functions are op overloads except for `_M_set_node`.

6.78.2 Member Function Documentation**`_M_set_node()`**

```
template<typename _Tp, typename _Ref, typename _Ptr>
void std::Deque_iterator<_Tp, _Ref, _Ptr>::_M_set_node (
    _Map_pointer __new_node) [inline], [noexcept]
```

Prepares to traverse `new_node`. Sets everything except `_M_cur`, which should therefore be set by the caller immediately afterwards, based on `_M_first` and `_M_last`.

The documentation for this struct was generated from the following files:

- [stl_algobase.h](#)
- [stl_deque.h](#)

6.79 `__gnu_parallel::_DRandomShufflingGlobalData<_RAIter>` Struct Template Reference

```
#include <random_shuffle.h>
```

Public Types

- typedef `_TraitsType::difference_type` `_DifferenceType`
- typedef `std::iterator_traits<_RAIter>` `_TraitsType`
- typedef `_TraitsType::value_type` `_ValueType`

Public Member Functions

- `_DRandomShufflingGlobalData` (`_RAIter &__source`)

Public Attributes

- `_ThreadIndex * _M_bin_proc`
- `_DifferenceType ** _M_dist`
- `int _M_num_bins`
- `int _M_num_bits`
- `_RAIter & _M_source`
- `_DifferenceType * _M_starts`
- `_ValueType ** _M_temporaries`

6.79.1 Detailed Description

```
template<typename _RAIter>
struct __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>
```

Data known to every thread participating in `__gnu_parallel::__parallel_random_shuffle()`.

6.79.2 Constructor & Destructor Documentation

`_DRandomShufflingGlobalData()`

```
template<typename _RAIter>
__gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_DRandomShufflingGlobalData (
    _RAIter & __source ) [inline]
```

Constructor.

6.79.3 Member Data Documentation

`_M_bin_proc`

```
template<typename _RAIter>
_ThreadIndex* __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_M_bin_proc
```

Number of the thread that will further process the corresponding bin.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__parallel_random_shuffle_drs_pu()`.

`_M_dist`

```
template<typename _RAIter >
```

```
_DifferenceType** __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_dist
```

Two-dimensional array to hold the thread-bin distribution.

Dimensions `(_M_num_threads + 1) __x (_M_num_bins + 1)`.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_num_bins`

```
template<typename _RAIter >
```

```
int __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_num_bins
```

Number of bins to distribute to.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_num_bits`

```
template<typename _RAIter >
```

```
int __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_num_bits
```

Number of bits needed to address the bins.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_source`

```
template<typename _RAIter >
```

```
_RAIter& __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_source
```

Begin iterator of the `__source`.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_starts`

```
template<typename _RAIter >
```

```
_DifferenceType* __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_starts
```

Start indexes of the threads' `__chunks`.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_temporaries`

```
template<typename _RAIter >
```

```
_ValueType** __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >::_M_temporaries
```

Temporary arrays for each thread.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

The documentation for this struct was generated from the following file:

- [random_shuffle.h](#)

6.80 `__gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>` Struct Template Reference

```
#include <random_shuffle.h>
```

Public Attributes

- [_BinIndex __bins_end](#)
- [_BinIndex _M_bins_begin](#)
- [int _M_num_threads](#)

- [_DRandomShufflingGlobalData<_RAIter> * _M_sd](#)
- [uint32_t _M_seed](#)

6.80.1 Detailed Description

```
template<typename _RAIter, typename _RandomNumberGenerator>
struct __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >
```

Local data for a thread participating in `__gnu_parallel::__parallel_random_shuffle()`.

6.80.2 Member Data Documentation

`__bins_end`

```
template<typename _RAIter , typename _RandomNumberGenerator >
_BinIndex __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >::__bins_end
```

End index for bins taken care of by this thread.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_bins_begin`

```
template<typename _RAIter , typename _RandomNumberGenerator >
_BinIndex __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >::_M_bins_begin
```

Begin index for bins taken care of by this thread.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_num_threads`

```
template<typename _RAIter , typename _RandomNumberGenerator >
int __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >::_M_num_threads
```

Number of threads participating in total.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_sd`

```
template<typename _RAIter , typename _RandomNumberGenerator >
_DRandomShufflingGlobalData<_RAIter>* __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >::_M_sd
```

Pointer to global data.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

`_M_seed`

```
template<typename _RAIter , typename _RandomNumberGenerator >
uint32_t __gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >::_M_seed
```

Random `_M_seed` for this thread.

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__parallel_random_shuffle_drs_pu\(\)](#).

The documentation for this struct was generated from the following file:

- [random_shuffle.h](#)

6.81 `__gnu_parallel::__DummyReduct` Struct Reference

```
#include <for_each_selectors.h>
```

Public Member Functions

- `bool operator() (bool, bool) const`

6.81.1 Detailed Description

Reduction function doing nothing.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.82 `__gnu_debug::_Equal_to<_Type>` Class Template Reference

```
#include <safe_sequence.h>
```

Public Member Functions

- `_Equal_to (const _Type &__v)`
- `bool operator() (const _Type &__x) const`

6.82.1 Detailed Description

```
template<typename _Type>
```

```
class __gnu_debug::_Equal_to<_Type>
```

A simple function object that returns true if the passed-in value is equal to the stored value.

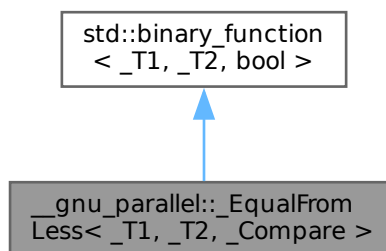
The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

6.83 `__gnu_parallel::_EqualFromLess<_T1, _T2, _Compare>` Class Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::_EqualFromLess<_T1, _T2, _Compare>`:

**Public Types**

- `typedef _T1 first_argument_type`
- `typedef bool result_type`
- `typedef _T2 second_argument_type`

Public Member Functions

- **_EqualFromLess** (*_Compare* &*__comp*)
- **bool operator()** (const *_T1* &*__a*, const *_T2* &*__b*)

6.83.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_EqualFromLess< _T1, _T2, _Compare >
```

Constructs predicate for equality from strict weak ordering predicate.

6.83.2 Member Typedef Documentation

first_argument_type

```
typedef _T1 std::binary_function< _T1 , _T2 , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary_function< _T1 , _T2 , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _T2 std::binary_function< _T1 , _T2 , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

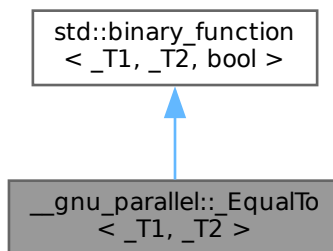
The documentation for this class was generated from the following file:

- [base.h](#)

6.84 __gnu_parallel::_EqualTo< _T1, _T2 > Struct Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::_EqualTo< _T1, _T2 >`:



Public Types

- typedef [_T1](#) [first_argument_type](#)
- typedef bool [result_type](#)
- typedef [_T2](#) [second_argument_type](#)

Public Member Functions

- bool **operator()** (const [_T1](#) &__t1, const [_T2](#) &__t2) const

6.84.1 Detailed Description

```
template<typename _T1, typename _T2>
struct __gnu_parallel::__EqualTo<_T1, _T2>
```

Similar to std::equal_to, but allows two different types.

6.84.2 Member Typedef Documentation

first_argument_type

```
typedef \_T1 std::binary\_function< \_T1 , \_T2 , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary\_function< \_T1 , \_T2 , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef \_T2 std::binary\_function< \_T1 , \_T2 , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [base.h](#)

6.85 std::__detail::__Executor<_Bilter, _Alloc, _TraitsT, __dfs_mode> Class Template Reference

```
#include <regex_executor.h>
```

Public Types

- typedef [iterator_traits](#)< [_Bilter](#) >::value_type [_CharT](#)
- typedef [_TraitsT](#)::char_class_type [_ClassT](#)
- typedef [regex_constants::match_flag_type](#) [_FlagT](#)
- typedef [_NFA](#)< [_TraitsT](#) > [_NFAT](#)
- typedef [basic_regex](#)< [_CharT](#), [_TraitsT](#) > [_RegexT](#)
- typedef [::vector](#)< [sub_match](#)< [_Bilter](#) >, [_Alloc](#) > [_ResultsVec](#)

Public Member Functions

- [_Executor](#) ([_Bilter](#) __begin, [_Bilter](#) __end, [_ResultsVec](#) &__results, const [_RegexT](#) &__re, [_FlagT](#) __flags)
- bool [_M_match](#) ()
- bool [_M_search](#) ()
- bool [_M_search_from_first](#) ()

Public Attributes

- `_Bilter _M_begin`
- `_ResultsVec _M_cur_results`
- `_Bilter _M_current`
- `const _Bilter _M_end`
- `_FlagT _M_flags`
- `bool _M_has_sol`
- `const _NFAT & _M_nfa`
- `const _RegexT & _M_re`
- `::vector< pair< _Bilter, int > > _M_rep_count`
- `_ResultsVec & _M_results`
- `_State_info< __search_mode, _ResultsVec > _M_states`

6.85.1 Detailed Description

```
template<typename _Bilter, typename _Alloc, typename _TraitsT, bool __dfs_mode>
class std::__detail::_Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode >
```

Takes a regex and an input string and does the matching.

The `_Executor` class has two modes: DFS mode and BFS mode, controlled by the template parameter `__dfs_mode`.

The documentation for this class was generated from the following files:

- [regex.h](#)
- [regex_executor.h](#)
- [regex_executor.tcc](#)

6.86 `__gnu_cxx::_ExtPtr_allocator< _Tp >` Class Template Reference

```
#include <extptr_allocator.h>
```

Public Types

- `typedef _Pointer_adapter< _Relative_pointer_impl< const _Tp > > const_pointer`
- `typedef const _Tp & const_reference`
- `typedef std::ptrdiff_t difference_type`
- `typedef _Pointer_adapter< _Relative_pointer_impl< _Tp > > pointer`
- `typedef _Tp & reference`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `_ExtPtr_allocator (const _ExtPtr_allocator &__rarg) noexcept`
- `template<typename _Up > _ExtPtr_allocator (const _ExtPtr_allocator< _Up > &__rarg) noexcept`
- `const std::allocator< _Tp > & _M_getUnderlyingImp () const`
- `const_pointer address (const_reference __x) const noexcept`
- `pointer address (reference __x) const noexcept`
- `pointer allocate (size_type __n, const void *__p=0)`
- `template<typename _Up, typename... _Args> void construct (_Up *__p, _Args &&... __args)`
- `template<typename... _Args> void construct (pointer __p, _Args &&... __args)`

- void **deallocate** ([pointer](#) __p, size_type __n)
- template<typename _Up >
void **destroy** (_Up *__p)
- void **destroy** ([pointer](#) __p)
- size_type **max_size** () const noexcept
- bool **operator==** (const [_ExtPtr_allocator](#) &__rarg) const
- template<typename _Up >
bool **operator==** (const [_ExtPtr_allocator](#)<_Up> &__rarg) const

Friends

- template<typename _Up >
void **swap** ([_ExtPtr_allocator](#)<_Up> &, [_ExtPtr_allocator](#)<_Up> &)

6.86.1 Detailed Description

template<typename _Tp>
class `__gnu_cxx::_ExtPtr_allocator<_Tp>`

An example allocator which uses a non-standard pointer type.

This allocator specifies that containers use a 'relative pointer' as it's pointer type. (See `ext/pointer.h`) Memory allocation in this example is still performed using `std::allocator`.

The documentation for this class was generated from the following file:

- [extptr_allocator.h](#)

6.87 `__gnu_cxx::__detail::_Ffit_finder<_Tp>` Class Template Reference

```
#include <bitmap_allocator.h>
```

Public Types

- typedef [_Block_pair](#) **argument_type**
- typedef bool **result_type**

Public Member Functions

- std::size_t * **_M_get** () const throw ()
- [_Counter_type](#) **_M_offset** () const throw ()
- bool **operator()** ([_Block_pair](#) __bp) throw ()

6.87.1 Detailed Description

template<typename _Tp>
class `__gnu_cxx::__detail::_Ffit_finder<_Tp>`

The class which acts as a predicate for applying the first-fit memory allocation policy for the bitmap allocator.

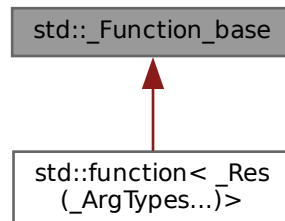
The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

6.88 std::_Function_base Class Reference

```
#include <std_function.h>
```

Inheritance diagram for std::_Function_base:



Public Types

- using **_Manager_type** = `bool(*)(_Any_data &, const _Any_data &, _Manager_operation)`

Public Member Functions

- `bool _M_empty () const`

Public Attributes

- `_Any_data _M_func`
- `_Manager_type _M_manager`

Static Public Attributes

- `static const size_t _M_max_align`
- `static const size_t _M_max_size`

6.88.1 Detailed Description

Base class of all polymorphic function object wrappers.

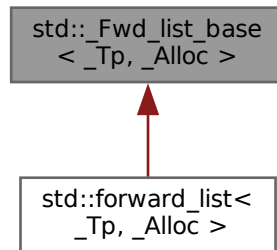
The documentation for this class was generated from the following file:

- [std_function.h](#)

6.89 std::_Fwd_list_base<_Tp, _Alloc > Struct Template Reference

```
#include <forward_list.h>
```

Inheritance diagram for std::_Fwd_list_base< _Tp, _Alloc >:



Public Types

- typedef [_Fwd_list_node](#)< _Tp > **_Node**
- typedef [_Fwd_list_const_iterator](#)< _Tp > **const_iterator**
- typedef [_Fwd_list_iterator](#)< _Tp > **iterator**

Public Member Functions

- **_Fwd_list_base** ([_Fwd_list_base](#) &&)=default
- **_Fwd_list_base** ([_Fwd_list_base](#) && __lst, [_Node_alloc_type](#) && __a)
- **_Fwd_list_base** ([_Fwd_list_base](#) && __lst, [_Node_alloc_type](#) && __a, [std::true_type](#))
- **_Fwd_list_base** ([_Node_alloc_type](#) && __a)
- const [_Node_alloc_type](#) & **_M_get_Node_allocator** () const noexcept
- [_Node_alloc_type](#) & **_M_get_Node_allocator** () noexcept

Protected Types

- typedef [__gnu_cxx::__alloc_traits](#)< [_Node_alloc_type](#) > **_Node_alloc_traits**
- typedef [__alloc_rebind](#)< [_Alloc](#), [_Fwd_list_node](#)< _Tp > > **_Node_alloc_type**

Protected Member Functions

- template<typename... _Args>
[_Node](#) * **_M_create_node** (_Args &&... __args)
- [_Fwd_list_node_base](#) * **_M_erase_after** ([_Fwd_list_node_base](#) * __pos)
- [_Fwd_list_node_base](#) * **_M_erase_after** ([_Fwd_list_node_base](#) * __pos, [_Fwd_list_node_base](#) * __last)
- [_Node](#) * **_M_get_node** ()
- template<typename... _Args>
[_Fwd_list_node_base](#) * **_M_insert_after** (const_iterator __pos, _Args &&... __args)
- void **_M_put_node** ([_Node](#) * __p)

Protected Attributes

- [_Fwd_list_impl](#) **_M_impl**

6.89.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
struct std::_Fwd_list_base< _Tp, _Alloc >
```

Base class for forward_list.

The documentation for this struct was generated from the following files:

- [forward_list.h](#)
- [forward_list.tcc](#)

6.90 std::_Fwd_list_const_iterator< _Tp > Struct Template Reference

```
#include <forward_list.h>
```

Public Types

- typedef const [_Fwd_list_node](#)< _Tp > **_Node**
- typedef [_Fwd_list_const_iterator](#)< _Tp > **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [_Fwd_list_iterator](#)< _Tp > **iterator**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef const _Tp * **pointer**
- typedef const _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_Fwd_list_const_iterator](#) (const [_Fwd_list_node_base](#) * __n) noexcept
- [_Fwd_list_const_iterator](#) (const [iterator](#) & __iter) noexcept
- [_Self](#) **_M_next** () const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- pointer **operator->** () const noexcept

Public Attributes

- const [_Fwd_list_node_base](#) * **_M_node**

Friends

- bool **operator==** (const [_Self](#) & __x, const [_Self](#) & __y) noexcept

6.90.1 Detailed Description

```
template<typename _Tp>
struct std::_Fwd_list_const_iterator< _Tp >
```

A forward_list::const_iterator.

All the functions are op overloads.

6.90.2 Friends And Related Symbol Documentation

operator==

```
template<typename _Tp >
bool operator== (
    const _Self & __x,
    const _Self & __y ) [friend]
```

Forward list const_iterator equality comparison.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

6.91 std::Fwd_list_iterator<_Tp> Struct Template Reference

```
#include <forward_list.h>
```

Public Types

- typedef [_Fwd_list_node](#)<_Tp> **_Node**
- typedef [_Fwd_list_iterator](#)<_Tp> **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef _Tp * **pointer**
- typedef _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_Fwd_list_iterator](#) ([_Fwd_list_node_base](#) *__n) noexcept
- [_Self](#) **_M_next** () const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- pointer **operator->** () const noexcept

Public Attributes

- [_Fwd_list_node_base](#) * **_M_node**

Friends

- bool [operator==](#) (const [_Self](#) &__x, const [_Self](#) &__y) noexcept

6.91.1 Detailed Description

```
template<typename _Tp>
struct std::Fwd_list_iterator<_Tp>
```

A forward_list::iterator.

All the functions are op overloads.

6.91.2 Friends And Related Symbol Documentation

operator==

```
template<typename _Tp >
bool operator== (
    const _Self & __x,
    const _Self & __y ) [friend]
```

Forward list iterator equality comparison.

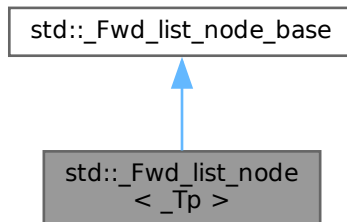
The documentation for this struct was generated from the following file:

- [forward_list.h](#)

6.92 std::_Fwd_list_node< _Tp > Struct Template Reference

```
#include <forward_list.h>
```

Inheritance diagram for std::_Fwd_list_node< _Tp >:



Public Member Functions

- void **_M_reverse_after** () noexcept
- [_Fwd_list_node_base](#) * **_M_transfer_after** ([_Fwd_list_node_base](#) * __begin, [_Fwd_list_node_base](#) * __end) noexcept
- const _Tp * **_M_valptr** () const noexcept
- _Tp * **_M_valptr** () noexcept

Public Attributes

- [_Fwd_list_node_base](#) * **_M_next**
- __gnu_cxx::__aligned_buffer< _Tp > **_M_storage**

6.92.1 Detailed Description

```
template<typename _Tp>
struct std::_Fwd_list_node< _Tp >
```

A helper node class for forward_list. This is just a linked list with uninitialized storage for a data value in each node. There is a sorting utility method.

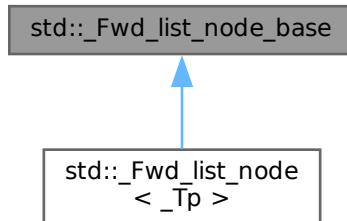
The documentation for this struct was generated from the following file:

- [forward_list.h](#)

6.93 std::_Fwd_list_node_base Struct Reference

```
#include <forward_list.h>
```

Inheritance diagram for std::_Fwd_list_node_base:



Public Member Functions

- [_Fwd_list_node_base](#) ([_Fwd_list_node_base](#) &&__x) noexcept
- [_Fwd_list_node_base](#) (const [_Fwd_list_node_base](#) &)=delete
- void [_M_reverse_after](#) () noexcept
- [_Fwd_list_node_base](#) * [_M_transfer_after](#) ([_Fwd_list_node_base](#) * __begin, [_Fwd_list_node_base](#) * __end) noexcept
- [_Fwd_list_node_base](#) & [operator=](#) ([_Fwd_list_node_base](#) &&__x) noexcept
- [_Fwd_list_node_base](#) & [operator=](#) (const [_Fwd_list_node_base](#) &)=delete

Public Attributes

- [_Fwd_list_node_base](#) * [_M_next](#)

6.93.1 Detailed Description

A helper basic node class for forward_list. This is just a linked list with nothing inside it. There are purely list shuffling utility methods here.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

6.94 __gnu_parallel::_GuardedIterator<_RAIter, _Compare> Class Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- [_GuardedIterator](#) ([_RAIter](#) __begin, [_RAIter](#) __end, [_Compare](#) &__comp)
- [operator _RAIter](#) () const
- [std::iterator_traits](#)<[_RAIter](#)>::value_type & [operator*](#) () const
- [_GuardedIterator](#)<[_RAIter](#), [_Compare](#)> & [operator++](#) ()

Friends

- bool `operator<` (const `_GuardedIterator`< `_RAIter`, `_Compare` > &__bi1, const `_GuardedIterator`< `_RAIter`, `_Compare` > &__bi2)
- bool `operator<=` (const `_GuardedIterator`< `_RAIter`, `_Compare` > &__bi1, const `_GuardedIterator`< `_RAIter`, `_Compare` > &__bi2)

6.94.1 Detailed Description

```
template<typename _RAIter, typename _Compare>
class __gnu_parallel::_GuardedIterator< _RAIter, _Compare >
```

`_Iterator` wrapper supporting an implicit supremum at the end of the sequence, dominating all comparisons.

The implicit supremum comes with a performance cost.

Deriving from `_RAIter` is not possible since `_RAIter` need not be a class.

6.94.2 Constructor & Destructor Documentation

`_GuardedIterator()`

```
template<typename _RAIter , typename _Compare >
__gnu_parallel::_GuardedIterator< _RAIter, _Compare >::_GuardedIterator (
    _RAIter __begin,
    _RAIter __end,
    _Compare & __comp ) [inline]
```

Constructor. Sets iterator to beginning of sequence.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator provided for associated overloaded compare operators.

6.94.3 Member Function Documentation

`operator _RAIter()`

```
template<typename _RAIter , typename _Compare >
__gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator _RAIter ( ) const [inline]
```

Convert to wrapped iterator.

Returns

Wrapped iterator.

`operator*()`

```
template<typename _RAIter , typename _Compare >
std::iterator_traits< _RAIter >::value_type & __gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator* ( ) const [inline]
```

Dereference operator.

Returns

Referenced element.

operator++()

```
template<typename _RAIter, typename _Compare >
__GuardedIterator< _RAIter, _Compare > & __gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator++ ( ) [inline]
```

Pre-increment operator.

Returns

This.

6.94.4 Friends And Related Symbol Documentation**operator<**

```
template<typename _RAIter, typename _Compare >
bool operator< (
    const __GuardedIterator< _RAIter, _Compare > & __bi1,
    const __GuardedIterator< _RAIter, _Compare > & __bi2 ) [friend]
```

Compare two elements referenced by guarded iterators.

Parameters

<code>__bi1</code>	First iterator.
<code>__bi2</code>	Second iterator.

Returns

true if less.

operator<=

```
template<typename _RAIter, typename _Compare >
bool operator<= (
    const __GuardedIterator< _RAIter, _Compare > & __bi1,
    const __GuardedIterator< _RAIter, _Compare > & __bi2 ) [friend]
```

Compare two elements referenced by guarded iterators.

Parameters

<code>__bi1</code>	First iterator.
<code>__bi2</code>	Second iterator.

Returns

True if less equal.

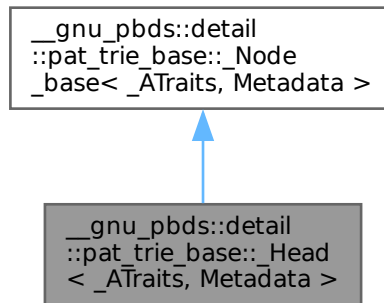
The documentation for this class was generated from the following file:

- [multiway_merge.h](#)

6.95 __gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata > Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata>`:



Public Types

- typedef `_ATraits::const_iterator` **a_const_iterator**
- typedef `detail::rebind_traits<_Alloc, _ATraits>::const_pointer` **a_const_pointer**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `_Node_base<_ATraits, Metadata>` **base_type**
- typedef `base_type::node_pointer` **node_pointer**
- typedef `base_type::type_traits` **type_traits**

Public Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_min**
- node_pointer **m_p_parent**
- const `node_type` **m_type**

6.95.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata>
```

Head node for PATRICIA tree.

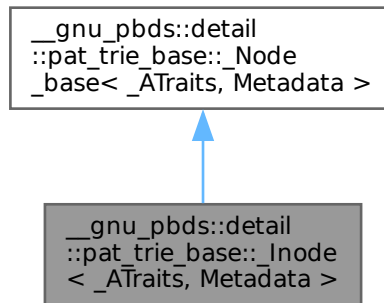
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.96 `__gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>` Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >:



Classes

- struct [const_iterator](#)
- struct [iterator](#)

Public Types

- enum { **arr_size** }
- typedef [detail::rebind_traits](#)<_Alloc, node_pointer > **__rebind_np**
- typedef base_type::allocator_type **_Alloc**
- typedef base_type::access_traits **access_traits**
- typedef _Alloc **allocator_type**
- typedef [_Node_base](#)<_ATraits, Metadata > **base_type**
- typedef __rebind_np::pointer **node_pointer_pointer**
- typedef __rebind_np::reference **node_pointer_reference**
- typedef _Alloc::size_type **size_type**
- typedef base_type::type_traits **type_traits**
- typedef type_traits::value_type **value_type**

Public Member Functions

- **_Inode** (size_type, const a_const_iterator)
- node_pointer **add_child** (node_pointer, a_const_iterator, a_const_iterator, a_const_pointer)
- [iterator](#) **begin** ()
- [const_iterator](#) **begin** () const
- [iterator](#) **end** ()
- [const_iterator](#) **end** () const
- [iterator](#) **get_child_it** (a_const_iterator, a_const_iterator, a_const_pointer)
- node_pointer **get_child_node** (a_const_iterator, a_const_iterator, a_const_pointer)
- node_const_pointer **get_child_node** (a_const_iterator, a_const_iterator, a_const_pointer) const
- size_type **get_e_ind** () const
- node_const_pointer **get_join_child** (node_const_pointer, a_const_pointer) const
- node_pointer **get_join_child** (node_pointer, a_const_pointer)

- node_pointer **get_lower_bound_child_node** (a_const_iterator, a_const_iterator, size_type, a_const_pointer)
- leaf_pointer **leftmost_descendant** ()
- leaf_const_pointer **leftmost_descendant** () const
- a_const_iterator **pref_b_it** () const
- a_const_iterator **pref_e_it** () const
- void **remove_child** (iterator)
- void **remove_child** (node_pointer)
- void **replace_child** (node_pointer, a_const_iterator, a_const_iterator, a_const_pointer)
- leaf_pointer **rightmost_descendant** ()
- leaf_const_pointer **rightmost_descendant** () const
- bool **should_be_mine** (a_const_iterator, a_const_iterator, size_type, a_const_pointer) const
- void **update_prefixes** (a_const_pointer)

Public Attributes

- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

6.96.1 Detailed Description

```
template<typename ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >
```

Internal node type, PATRICIA tree.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.97 __gnu_cxx::_Invalid_type Struct Reference

```
#include <pointer.h>
```

6.97.1 Detailed Description

The specialization on this type helps resolve the problem of reference to void, and eliminates the need to specialize `_Pointer_adapter` for cases of `void*`, `const void*`, and so on.

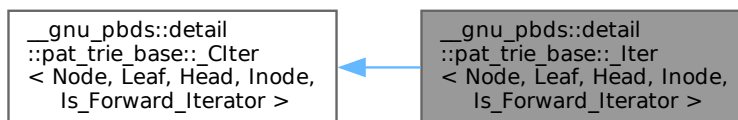
The documentation for this struct was generated from the following file:

- [pointer.h](#)

6.98 __gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator > Class Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`:



Public Types

- typedef allocator_type **_Alloc**
- typedef base_type::allocator_type **allocator_type**
- typedef [_CIter](#)< Node, Leaf, Head, Inode, Is_Forward_Iterator > **base_type**
- typedef type_traits::const_pointer **const_pointer**
- typedef type_traits::const_reference **const_reference**
- typedef allocator_type::difference_type **difference_type**
- typedef base_type::head_pointer **head_pointer**
- typedef Inode::iterator **inode_iterator**
- typedef base_type::inode_pointer **inode_pointer**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef base_type::leaf_const_pointer **leaf_const_pointer**
- typedef base_type::leaf_pointer **leaf_pointer**
- typedef base_type::node_pointer **node_pointer**
- typedef type_traits::pointer **pointer**
- typedef type_traits::reference **reference**
- typedef base_type::type_traits **type_traits**
- typedef type_traits::value_type **value_type**

Public Member Functions

- [_Iter](#) (const [_Iter](#)< Node, Leaf, Head, Inode, Is_Forward_Iterator > &other)
- [_Iter](#) (node_pointer p_nd=0)
- bool **operator!=** (const [_CIter](#) &other) const
- bool **operator!=** (const [_CIter](#)< Node, Leaf, Head, Inode, Is_Forward_Iterator > &other) const
- reference **operator*** () const
- [_Iter](#) & **operator++** ()
- [_Iter](#) **operator++** (int)
- [_Iter](#) & **operator--** ()
- [_Iter](#) **operator--** (int)
- pointer **operator->** () const
- [_Iter](#) & **operator=** (const [_Iter](#) &other)
- [_Iter](#) & **operator=** (const [_Iter](#)< Node, Leaf, Head, Inode, Is_Forward_Iterator > &other)
- bool **operator==** (const [_CIter](#) &other) const
- bool **operator==** (const [_CIter](#)< Node, Leaf, Head, Inode, Is_Forward_Iterator > &other) const

Public Attributes

- node_pointer **m_p_nd**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

Static Protected Member Functions

- static node_pointer **get_larger_sibling** (node_pointer p_nd)
- static node_pointer **get_smaller_sibling** (node_pointer p_nd)
- static leaf_pointer **leftmost_descendant** (node_pointer p_nd)
- static leaf_pointer **rightmost_descendant** (node_pointer p_nd)

6.98.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, bool Is_Forward_Iterator>
class __gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >
```

Iterator.

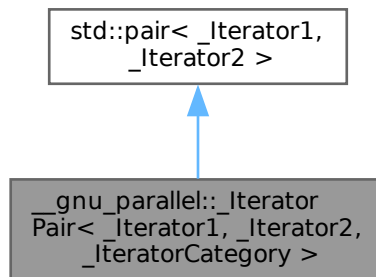
The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

6.99 __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory > Class Template Reference

```
#include <iterator.h>
```

Inheritance diagram for __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >:



Public Types

- typedef [std::iterator_traits](#)< _Iterator1 > **_TraitsType**
- typedef _TraitsType::difference_type **difference_type**
- typedef _Iterator1 [first_type](#)
- typedef _IteratorCategory **iterator_category**
- typedef [_IteratorPair](#) * **pointer**
- typedef [_IteratorPair](#) & **reference**
- typedef _Iterator2 [second_type](#)
- typedef void **value_type**

Public Member Functions

- [_IteratorPair](#) (const _Iterator1 &__first, const _Iterator2 &__second)
- **operator _Iterator2** () const
- [_IteratorPair](#) **operator+** (difference_type __delta) const
- [_IteratorPair](#) & **operator++** ()
- const [_IteratorPair](#) **operator++** (int)
- difference_type **operator-** (const [_IteratorPair](#) &__other) const
- [_IteratorPair](#) & **operator--** ()
- const [_IteratorPair](#) **operator--** (int)

- `_IteratorPair` & **operator=** (const `_IteratorPair` &__other)
- constexpr void **swap** (`pair` &__p) noexcept(__and_< __is_nothrow_swappable<_Iterator1>, __is_nothrow_swappable<_Iterator2>>::value)

Public Attributes

- `_Iterator1` [first](#)
- `_Iterator2` [second](#)

Related Symbols

(Note that these are not member symbols.)

- constexpr `pair<typename __decay_and_strip<_Iterator1>::__type, typename __decay_and_strip<_Iterator2>::__type>` [make_pair](#) (`_Iterator1` &&__x, `_Iterator2` &&__y)
- constexpr enable_if< __and_< __is_swappable<_Iterator1>, __is_swappable<_Iterator2>>::value >::type [swap](#) (`pair<_Iterator1, _Iterator2>` &__x, `pair<_Iterator1, _Iterator2>` &__y) noexcept(noexcept(__x.swap(__y)))
- constexpr bool [operator==](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)
- constexpr bool [operator<](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)
- constexpr bool [operator!=](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)
- constexpr bool [operator>](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)
- constexpr bool [operator<=](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)
- constexpr bool [operator>=](#) (const `pair<_Iterator1, _Iterator2>` &__x, const `pair<_Iterator1, _Iterator2>` &__y)

6.99.1 Detailed Description

```
template<typename _Iterator1, typename _Iterator2, typename _IteratorCategory>
class __gnu_parallel::__IteratorPair<_Iterator1, _Iterator2, _IteratorCategory>
```

A pair of iterators. The usual iterator operations are applied to both child iterators.

6.99.2 Member Typedef Documentation

`first_type`

```
typedef _Iterator1 std::pair<\_Iterator1, \_Iterator2>::first\_type [inherited]
```

The type of the `first` member.

`second_type`

```
typedef _Iterator2 std::pair<\_Iterator1, \_Iterator2>::second\_type [inherited]
```

The type of the `second` member.

6.99.3 Member Function Documentation

`swap()`

```
constexpr void std::pair<\_Iterator1, \_Iterator2>::swap (
    pair<\_Iterator1, \_Iterator2> &__p) [inline], [constexpr], [noexcept], [inherited]
```

Swap the first members and then the second members.

6.99.4 Friends And Related Symbol Documentation

make_pair()

```
constexpr pair< typename __decay_and_strip< _Iterator1 >::__type, typename __decay_and_strip< _Iterator2 >::__type > make_pair (
    _Iterator1 && __x,
    _Iterator2 && __y ) [related]
```

A convenience wrapper for creating a pair from two objects.

Parameters

\leftrightarrow __x	The first object.
\leftrightarrow __y	The second object.

Returns

A newly-constructed pair<> object of the appropriate type.

The C++98 standard says the objects are passed by reference-to-const, but C++03 says they are passed by value (this was LWG issue #181).

Since C++11 they have been passed by forwarding reference and then forwarded to the new members of the pair. To create a pair with a member of reference type, pass a `reference_wrapper` to this function.

operator"!=()

```
constexpr bool operator!= (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Uses `operator==` to find the result.

operator<()

```
constexpr bool operator< (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Defines a lexicographical order for pairs.

For two pairs of the same type, `P` is ordered before `Q` if `P.first` is less than `Q.first`, or if `P.first` and `Q.first` are equivalent (neither is less than the other) and `P.second` is less than `Q.second`.

operator<=()

```
constexpr bool operator<= (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Uses `operator<` to find the result.

operator==(())

```
constexpr bool operator== (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Two pairs of the same type are equal iff their members are equal.

operator>()

```
constexpr bool operator> (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Uses `operator<` to find the result.

operator>=()

```
constexpr bool operator>= (
    const pair< _Iterator1 , _Iterator2 > & __x,
    const pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Uses `operator<` to find the result.

swap()

```
constexpr enable_if< __and< __is_swappable< _Iterator1 >, __is_swappable< _Iterator2 > >↵
::value >::type swap (
    pair< _Iterator1 , _Iterator2 > & __x,
    pair< _Iterator1 , _Iterator2 > & __y ) [related]
```

Swap overload for pairs. Calls `std::pair::swap()`.

Note

This `std::swap` overload is not declared in C++03 mode, which has performance implications, e.g. see <https://gcc.gnu.org/PR38466>

6.99.5 Member Data Documentation**first**

```
_Iterator1 std::pair< _Iterator1 , _Iterator2 >::first [inherited]
```

The first member.

second

```
_Iterator2 std::pair< _Iterator1 , _Iterator2 >::second [inherited]
```

The second member.

The documentation for this class was generated from the following file:

- [iterator.h](#)

6.100 `__gnu_parallel::_IteratorTriple<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory >` Class Template Reference

```
#include <iterator.h>
```

Public Types

- typedef `std::iterator_traits<_Iterator1 >::difference_type` **difference_type**
- typedef `_IteratorCategory` **iterator_category**
- typedef `_IteratorTriple *` **pointer**
- typedef `_IteratorTriple &` **reference**
- typedef void **value_type**

Public Member Functions

- **_IteratorTriple** (const _Iterator1 &__first, const _Iterator2 &__second, const _Iterator3 &__third)
- **operator _Iterator3** () const
- **_IteratorTriple operator+** (difference_type __delta) const
- **_IteratorTriple & operator++** ()
- const **_IteratorTriple operator++** (int)
- difference_type **operator-** (const _IteratorTriple &__other) const
- **_IteratorTriple & operator--** ()
- const **_IteratorTriple operator--** (int)
- **_IteratorTriple & operator=** (const _IteratorTriple &__other)

Public Attributes

- _Iterator1 **_M_first**
- _Iterator2 **_M_second**
- _Iterator3 **_M_third**

6.100.1 Detailed Description

```
template<typename _Iterator1, typename _Iterator2, typename _Iterator3, typename _IteratorCategory>
class __gnu_parallel::_IteratorTriple< _Iterator1, _Iterator2, _Iterator3, _IteratorCategory >
```

A triple of iterators. The usual iterator operations are applied to all three child iterators.

The documentation for this class was generated from the following file:

- [iterator.h](#)

6.101 __gnu_parallel::_Job< _DifferenceTp > Struct Template Reference

```
#include <workstealing.h>
```

Public Types

- typedef _DifferenceTp **_DifferenceType**

Public Attributes

- volatile _DifferenceType **_M_first**
- volatile _DifferenceType **_M_last**
- volatile _DifferenceType **_M_load**

6.101.1 Detailed Description

```
template<typename _DifferenceTp>
struct __gnu_parallel::_Job< _DifferenceTp >
```

One __job for a certain thread.

6.101.2 Member Data Documentation

_M_first

```
template<typename _DifferenceTp >
volatile _DifferenceType __gnu_parallel::_Job< _DifferenceTp >::_M_first
```

First element.

Changed by owning and stealing thread. By stealing thread, always incremented.

Referenced by [__gnu_parallel::__for_each_template_random_access_workstealing\(\)](#).

_M_last

```
template<typename _DifferenceTp >
volatile _DifferenceType __gnu_parallel::_Job< _DifferenceTp >::_M_last
```

Last element.
 Changed by owning thread only.
 Referenced by `__gnu_parallel::_for_each_template_random_access_workstealing()`.

_M_load

```
template<typename _DifferenceTp >
volatile _DifferenceType __gnu_parallel::_Job< _DifferenceTp >::_M_load
```

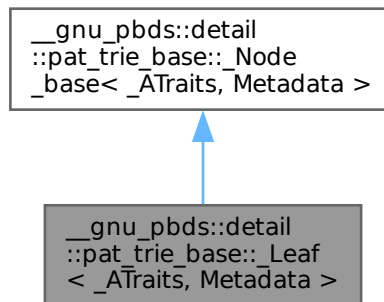
Number of elements, i.e. `_M_last - _M_first + 1`.
 Changed by owning thread only.
 Referenced by `__gnu_parallel::_for_each_template_random_access_workstealing()`.
 The documentation for this struct was generated from the following file:

- [workstealing.h](#)

6.102 `__gnu_pbds::detail::pat_trie_base::_Leaf<_ATraits, Metadata>` Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Leaf<_ATraits, Metadata>`:

**Public Types**

- typedef `_ATraits::const_iterator` **a_const_iterator**
- typedef `detail::rebind_traits<_Alloc, _ATraits>::const_pointer` **a_const_pointer**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `_Node_base<_ATraits, Metadata>` **base_type**
- typedef `type_traits::const_reference` **const_reference**
- typedef `detail::rebind_traits<_Alloc, _Node_base>::pointer` **node_pointer**
- typedef `type_traits::reference` **reference**
- typedef `base_type::type_traits` **type_traits**
- typedef `type_traits::value_type` **value_type**

Public Member Functions

- **_Leaf** (const_reference other)
- reference **value** ()
- const_reference **value** () const

Public Attributes

- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

6.102.1 Detailed Description

```
template<typename ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Leaf< ATraits, Metadata >
```

Leaf node for PATRICIA tree.

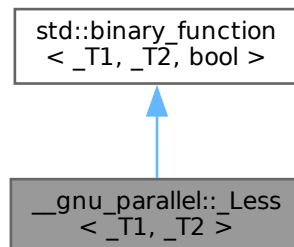
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.103 __gnu_parallel::_Less<_T1, _T2 > Struct Template Reference

```
#include <base.h>
```

Inheritance diagram for __gnu_parallel::_Less<_T1, _T2>:



Public Types

- typedef `_T1` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_T2` [second_argument_type](#)

Public Member Functions

- `bool` **operator()** (const `_T1` &__t1, const `_T2` &__t2) const
- `bool` **operator()** (const `_T2` &__t2, const `_T1` &__t1) const

6.103.1 Detailed Description

```
template<typename _T1, typename _T2>
struct __gnu_parallel::_Less< _T1, _T2 >
```

Similar to `std::less`, but allows two different types.

6.103.2 Member Typedef Documentation

first_argument_type

```
typedef _T1 std::binary_function< _T1 , _T2 , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary_function< _T1 , _T2 , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _T2 std::binary_function< _T1 , _T2 , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

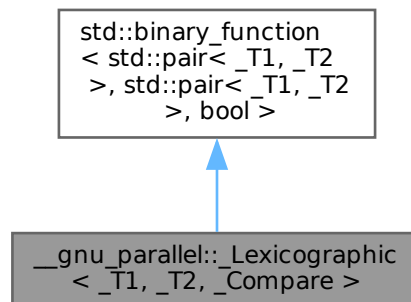
The documentation for this struct was generated from the following file:

- [base.h](#)

6.104 __gnu_parallel::_Lexicographic< _T1, _T2, _Compare > Class Template Reference

```
#include <multiseq_selection.h>
```

Inheritance diagram for `__gnu_parallel::_Lexicographic< _T1, _T2, _Compare >`:



Public Types

- typedef `std::pair< _T1, _T2 >` `first_argument_type`
- typedef `bool` `result_type`
- typedef `std::pair< _T1, _T2 >` `second_argument_type`

Public Member Functions

- **_Lexicographic** (_Compare &__comp)
- **bool operator()** (const [std::pair](#)< _T1, _T2 > &__p1, const [std::pair](#)< _T1, _T2 > &__p2) const

6.104.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_Lexicographic< _T1, _T2, _Compare >
```

Compare __a pair of types lexicographically, ascending.

6.104.2 Member Typedef Documentation

first_argument_type

```
typedef std::pair< _T1, _T2 > std::binary\_function< std::pair< _T1, _T2 > , std::pair< _T1, _T2 > , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary\_function< std::pair< _T1, _T2 > , std::pair< _T1, _T2 > , bool >↵::result_type [inherited]
result_type is the return type
```

second_argument_type

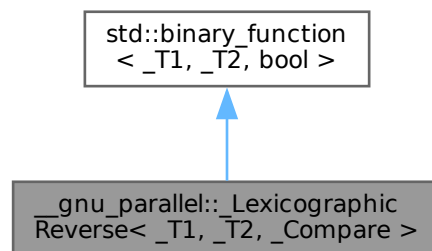
```
typedef std::pair< _T1, _T2 > std::binary\_function< std::pair< _T1, _T2 > , std::pair< _T1, _T2 > , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
The documentation for this class was generated from the following file:
```

- [multiseq_selection.h](#)

6.105 __gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare > Class Template Reference

```
#include <multiseq_selection.h>
```

Inheritance diagram for __gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare >:



Public Types

- typedef `_T1` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_T2` [second_argument_type](#)

Public Member Functions

- `_LexicographicReverse` (`_Compare &__comp`)
- `bool operator()` (`const std::pair<_T1, _T2> &__p1, const std::pair<_T1, _T2> &__p2`) `const`

6.105.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_LexicographicReverse<_T1, _T2, _Compare>
```

Compare __a pair of types lexicographically, descending.

6.105.2 Member Typedef Documentation**first_argument_type**

```
typedef _T1 std::binary_function<_T1, _T2, bool>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary_function<_T1, _T2, bool>::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _T2 std::binary_function<_T1, _T2, bool>::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

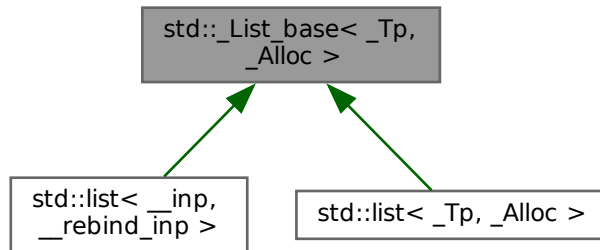
The documentation for this class was generated from the following file:

- [multiseq_selection.h](#)

6.106 `std::_List_base<_Tp, _Alloc>` Class Template Reference

```
#include <stl_list.h>
```

Inheritance diagram for `std::_List_base<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc allocator_type`

Public Member Functions

- `_List_base` (`_List_base` &&)=default
- `_List_base` (`_List_base` &&_x, `_Node_alloc_type` &&_a)
- `_List_base` (`_Node_alloc_type` &&_a)
- `_List_base` (`_Node_alloc_type` &&_a, `_List_base` &&_x)
- `_List_base` (`const _Node_alloc_type` &_a) noexcept
- `void _M_clear` () noexcept
- `const _Node_alloc_type & _M_get_Node_allocator` () const noexcept
- `_Node_alloc_type & _M_get_Node_allocator` () noexcept
- `void _M_init` () noexcept
- `void _M_move_nodes` (`_List_base` &&_x)

Protected Types

- typedef `__gnu_cxx::__alloc_traits<_Node_alloc_type> _Node_alloc_traits`
- typedef `_Tp_alloc_traits::template rebind<_List_node<_Tp>>::other _Node_alloc_type`
- typedef `__gnu_cxx::__alloc_traits<_Tp_alloc_type> _Tp_alloc_traits`
- typedef `__gnu_cxx::__alloc_traits<_Alloc>::template rebind<_Tp>::other _Tp_alloc_type`

Protected Member Functions

- `void _M_dec_size` (`size_t`)
- `size_t _M_distance` (`const void *`, `const void *`) const
- `_Node_alloc_traits::pointer _M_get_node` ()
- `size_t _M_get_size` () const
- `void _M_inc_size` (`size_t`)
- `size_t _M_node_count` () const
- `void _M_put_node` (`typename _Node_alloc_traits::pointer __p`) noexcept
- `void _M_set_size` (`size_t`)

Static Protected Member Functions

- static `size_t` **S_distance** (const `__detail::_List_node_base *`__first, const `__detail::_List_node_base *`__last)

Protected Attributes

- `_List_impl` `_M_impl`

6.106.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
class std::_List_base< _Tp, _Alloc >
```

See `bits/stl_deque.h`'s `_Deque_base` for an explanation.

The documentation for this class was generated from the following files:

- [stl_list.h](#)
- [list.tcc](#)

6.107 `std::_List_const_iterator< _Tp >` Struct Template Reference

```
#include <stl_list.h>
```

Public Types

- typedef const `_List_node`< `_Tp` > `_Node`
- typedef `_List_const_iterator`< `_Tp` > `_Self`
- typedef `ptrdiff_t` **difference_type**
- typedef `_List_iterator`< `_Tp` > **iterator**
- typedef `std::bidirectional_iterator_tag` **iterator_category**
- typedef const `_Tp *` **pointer**
- typedef const `_Tp &` **reference**
- typedef `_Tp` **value_type**

Public Member Functions

- `_List_const_iterator` (const `__detail::_List_node_base *`__x) noexcept
- `_List_const_iterator` (const `iterator` &__x) noexcept
- `iterator` **_M_const_cast** () const noexcept
- reference **operator*** () const noexcept
- `_Self` & **operator++** () noexcept
- `_Self` **operator++** (int) noexcept
- `_Self` & **operator--** () noexcept
- `_Self` **operator--** (int) noexcept
- pointer **operator->** () const noexcept

Public Attributes

- const `__detail::_List_node_base *` **_M_node**

Friends

- bool **operator==** (const `_Self` &__x, const `_Self` &__y) noexcept

6.107.1 Detailed Description

```
template<typename _Tp>
struct std::_List_const_iterator< _Tp >
```

A list::const_iterator.

All the functions are op overloads.

The documentation for this struct was generated from the following files:

- [stl_iterator_base_funcs.h](#)
- [stl_list.h](#)

6.108 std::_List_iterator< _Tp > Struct Template Reference

```
#include <stl_list.h>
```

Public Types

- typedef [_List_node](#)<_Tp> **_Node**
- typedef [_List_iterator](#)<_Tp> **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef _Tp * **pointer**
- typedef _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_List_iterator](#) ([__detail::_List_node_base](#) *__x) noexcept
- [_Self_M_const_cast](#) () const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- [_Self](#) & **operator--** () noexcept
- [_Self](#) **operator--** (int) noexcept
- pointer **operator->** () const noexcept

Public Attributes

- [__detail::_List_node_base](#) * **_M_node**

Friends

- bool **operator==** (const [_Self](#) &__x, const [_Self](#) &__y) noexcept

6.108.1 Detailed Description

```
template<typename _Tp>
struct std::_List_iterator< _Tp >
```

A list::iterator.

All the functions are op overloads.

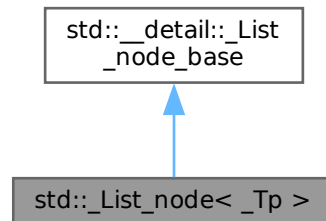
The documentation for this struct was generated from the following files:

- [stl_iterator_base_funcs.h](#)
- [stl_list.h](#)

6.109 `std::_List_node<_Tp>` Struct Template Reference

```
#include <stl_list.h>
```

Inheritance diagram for `std::_List_node<_Tp>`:



Public Member Functions

- `void _M_hook (_List_node_base *const __position) noexcept`
- `void _M_reverse () noexcept`
- `void _M_transfer (_List_node_base *const __first, _List_node_base *const __last) noexcept`
- `void _M_unhook () noexcept`
- `_Tp * _M_valptr ()`
- `_Tp const * _M_valptr () const`

Static Public Member Functions

- `static void swap (_List_node_base &__x, _List_node_base &__y) noexcept`

Public Attributes

- `_List_node_base * _M_next`
- `_List_node_base * _M_prev`
- `__gnu_cxx::__aligned_membuf<_Tp> _M_storage`

6.109.1 Detailed Description

```
template<typename _Tp>
struct std::_List_node<_Tp>
```

An actual node in the list.

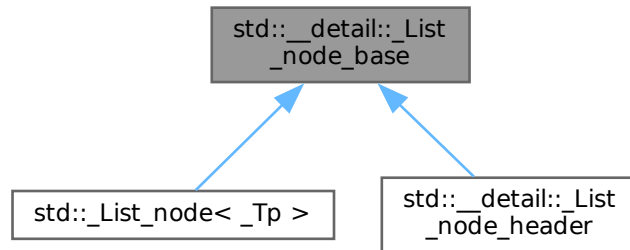
The documentation for this struct was generated from the following file:

- [stl_list.h](#)

6.110 `std::__detail::_List_node_base` Struct Reference

```
#include <stl_list.h>
```


Inheritance diagram for `std::__detail::_List_node_base`:



Public Member Functions

- `void _M_hook (_List_node_base *const __position) noexcept`
- `void _M_reverse () noexcept`
- `void _M_transfer (_List_node_base *const __first, _List_node_base *const __last) noexcept`
- `void _M_unhook () noexcept`

Static Public Member Functions

- `static void swap (_List_node_base &__x, _List_node_base &__y) noexcept`

Public Attributes

- `_List_node_base * _M_next`
- `_List_node_base * _M_prev`

6.110.1 Detailed Description

Common part of a node in the list.

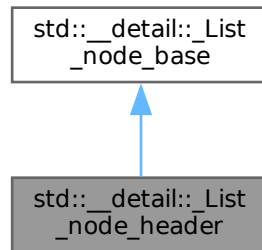
The documentation for this struct was generated from the following file:

- [stl_list.h](#)

6.111 `std::__detail::_List_node_header` Struct Reference

```
#include <stl_list.h>
```

Inheritance diagram for `std::__detail::_List_node_header`:



Public Member Functions

- `_List_node_header` (`_List_node_header` &&__x) noexcept
- `void _M_hook` (`_List_node_base` *const __position) noexcept
- `void _M_init` () noexcept
- `void _M_move_nodes` (`_List_node_header` &&__x)
- `void _M_reverse` () noexcept
- `void _M_transfer` (`_List_node_base` *const __first, `_List_node_base` *const __last) noexcept
- `void _M_unhook` () noexcept

Static Public Member Functions

- `static void swap` (`_List_node_base` &__x, `_List_node_base` &__y) noexcept

Public Attributes

- `_List_node_base` * `_M_next`
- `_List_node_base` * `_M_prev`

6.111.1 Detailed Description

The list node header.

The documentation for this struct was generated from the following file:

- [stl_list.h](#)

6.112 `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser` Struct Reference

```
#include <losertree.h>
```

Public Attributes

- `_Tp` `_M_key`
- `int` `_M_source`
- `bool` `_M_sup`

6.112.1 Detailed Description

```
template<typename _Tp, typename _Compare>
struct __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser
```

Internal representation of a `_LoserTree` element.

6.112.2 Member Data Documentation

`_M_key`

```
template<typename _Tp , typename _Compare >
_Tp __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_key
_M_key of the element in the _LoserTree.
Referenced by \_\_gnu\_parallel::\_LoserTreeBase< \_Tp, \_Compare >::\_\_insert\_start\(\).
```

`_M_source`

```
template<typename _Tp , typename _Compare >
int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_source
_index of the __source __sequence.
Referenced by \_\_gnu\_parallel::\_LoserTreeBase< \_Tp, \_Compare >::\_\_get\_min\_source\(\), and \_\_gnu\_parallel::\_LoserTreeBase< \_Tp, \_Compare >::\_\_insert\_start\(\).
```

`_M_sup`

```
template<typename _Tp , typename _Compare >
bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_sup
flag, true iff this is a "maximum" __sentinel.
Referenced by \_\_gnu\_parallel::\_LoserTreeBase< \_Tp, \_Compare >::\_\_insert\_start\(\).
The documentation for this struct was generated from the following file:
```

- [losertree.h](#)

6.113 `__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser` Struct Reference

```
#include <losertree.h>
```

Public Attributes

- `const _Tp * _M_keyp`
- `int _M_source`
- `bool _M_sup`

6.113.1 Detailed Description

```
template<typename _Tp, typename _Compare>
struct __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser
```

Internal representation of `_LoserTree` `__elements`.

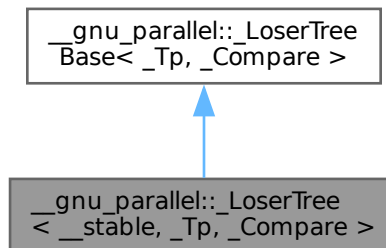
The documentation for this struct was generated from the following file:

- [losertree.h](#)

6.114 `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >` Class Template Reference

```
#include <loserTree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >`:

**Public Member Functions**

- `_LoserTree` (unsigned int `__k`, `_Compare` `__comp`)
- void `__delete_min_insert` (`_Tp` `__key`, bool `__sup`)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int `__root`)
- void `__insert_start` (const `_Tp` &`__key`, int `__source`, bool `__sup`)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_log_k`
- unsigned int `_M_offset`

6.114.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >
```

Stable `_LoserTree` variant.

Provides the stable implementations of `insert_start`, `__init_winner`, `__init` and `__delete_min_insert`.

Unstable variant is done using partial specialisation below.

6.114.2 Member Function Documentation**`__delete_min_insert()`**

```
template<bool __stable, typename _Tp, typename _Compare >
void __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::__delete_min_insert (
    _Tp __key,
    bool __sup ) [inline]
```

Delete the smallest element and insert a new element from the previously smallest element's sequence.

This implementation is stable.

References [std::swap\(\)](#).

__get_min_source()

```
template<typename _Tp , typename _Compare >
int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__get_min_source ( ) [inline], [inherited]
```

Returns

the index of the sequence with the smallest element.

References [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_losers](#), and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::](#)

__insert_start()

```
template<typename _Tp , typename _Compare >
void __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__insert_start (
    const _Tp & __key,
    int __source,
    bool __sup ) [inline], [inherited]
```

Initializes the sequence “_M_source” with the element “__key”.

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	<code>__index</code> of the <code>__source</code> sequence
<code>__sup</code>	flag that determines whether the value to insert is an explicit <code>__supremum</code> .

References [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_first_insert](#), [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::](#)
[__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_losers](#), [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::__M_sou](#)
and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::__M_sup](#).

6.114.3 Member Data Documentation**__M_log_k**

```
template<typename _Tp , typename _Compare >
unsigned int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_log_k [protected], [inherited]
log_2{__M_k}
```

Referenced by [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#).

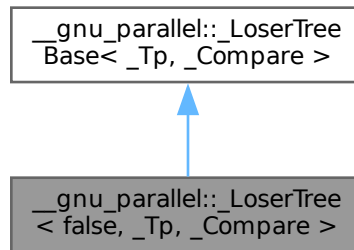
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.115 __gnu_parallel::_LoserTree< false, _Tp, _Compare > Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTree< false, _Tp, _Compare >`:



Public Member Functions

- `_LoserTree` (unsigned int `__k`, `_Compare` `__comp`)
- void `__delete_min_insert` (`_Tp` `__key`, bool `__sup`)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int `__root`)
- void `__insert_start` (const `_Tp` & `__key`, int `__source`, bool `__sup`)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_offset`

6.115.1 Detailed Description

template<typename `_Tp`, typename `_Compare`>
class `__gnu_parallel::_LoserTree< false, _Tp, _Compare >`

Unstable `_LoserTree` variant.

Stability (non-stable here) is selected with partial specialization.

6.115.2 Member Function Documentation

`__delete_min_insert()`

```

template<typename _Tp , typename _Compare >
void __gnu_parallel::_LoserTree< false, _Tp, _Compare >::__delete_min_insert (
    _Tp __key,
    bool __sup ) [inline]
  
```

Delete the `_M_key` smallest element and insert the element `__key` instead.

Parameters

<code>__key</code>	the <code>_M_key</code> to insert
<code>__sup</code>	true iff <code>__key</code> is an explicitly marked supremum

References [std::swap\(\)](#).

__get_min_source()

```
template<typename _Tp , typename _Compare >
int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__get_min_source ( ) [inline], [inherited]
```

Returns

the index of the sequence with the smallest element.

References [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::M_losers](#), and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::](#)

__init_winner()

```
template<typename _Tp , typename _Compare >
unsigned int __gnu_parallel::_LoserTree< false, _Tp, _Compare >::__init_winner (
    unsigned int __root ) [inline]
```

Computes the winner of the competition at position “__root”.

Called recursively (starting at 0) to build the initial tree.

Parameters

<code>__root</code>	__index of the “game” to start.
---------------------	---------------------------------

__insert_start()

```
template<typename _Tp , typename _Compare >
void __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__insert_start (
    const _Tp & __key,
    int __source,
    bool __sup ) [inline], [inherited]
```

Initializes the sequence “_M_source” with the element “__key”.

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	__index of the __source __sequence
<code>__sup</code>	flag that determines whether the value to insert is an explicit __supremum.

References [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::M_first_insert](#), [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::](#)
[__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::M_losers](#), [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_sou](#)
and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_sup](#).

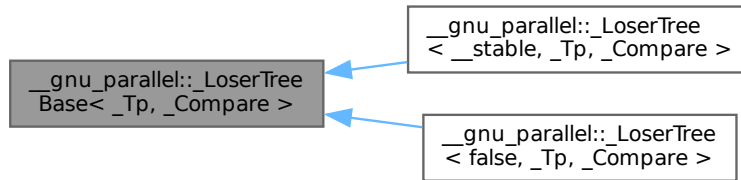
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.116 __gnu_parallel::_LoserTreeBase< _Tp, _Compare > Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>`:



Classes

- struct `_Loser`

Public Member Functions

- `_LoserTreeBase` (unsigned int __k, _Compare __comp)
- `~_LoserTreeBase` ()
- int `__get_min_source` ()
- void `__insert_start` (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- _Compare `_M_comp`
- bool `_M_first_insert`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- unsigned int `_M_log_k`
- `_Loser` * `_M_losers`
- unsigned int `_M_offset`

6.116.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeBase<_Tp, _Compare>
```

Guarded loser/tournament tree.

The smallest element is at the top.

Guarding is done explicitly through one flag `_M_sup` per element, `inf` is not needed due to a better initialization routine.

This is a well-performing variant.

Parameters

<code>_Tp</code>	the element type
<code>_Compare</code>	the comparator to use, defaults to <code>std::less<_Tp></code>

6.116.2 Constructor & Destructor Documentation

`__LoserTreeBase()`

```
template<typename _Tp , typename _Compare >
__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__LoserTreeBase (
    unsigned int __k,
    _Compare __comp ) [inline]
```

The constructor.

Parameters

<code>__k</code>	The number of sequences to merge.
<code>__comp</code>	The comparator to use.

References [__gnu_parallel::__rd_log2\(\)](#), [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_first_insert](#), [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_log_k](#), and [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_losers](#).

`~__LoserTreeBase()`

```
template<typename _Tp , typename _Compare >
__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::~~__LoserTreeBase ( ) [inline]
```

The destructor.

References [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_losers](#).

6.116.3 Member Function Documentation

`__get_min_source()`

```
template<typename _Tp , typename _Compare >
int __gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__get_min_source ( ) [inline]
```

Returns

the index of the sequence with the smallest element.

References [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_losers](#), and [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_sup](#).

`__insert_start()`

```
template<typename _Tp , typename _Compare >
void __gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__insert_start (
    const _Tp & __key,
    int __source,
    bool __sup ) [inline]
```

Initializes the sequence “`_M_source`” with the element “`__key`”.

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	<code>__index</code> of the <code>__source</code> sequence
<code>__sup</code>	flag that determines whether the value to insert is an explicit <code>__supremum</code> .

References [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_first_insert](#), [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__M_losers](#), [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__Loser::__M_source](#), and [__gnu_parallel::__LoserTreeBase< _Tp, _Compare >::__Loser::__M_sup](#).

6.116.4 Member Data Documentation

`_M_comp`

```
template<typename _Tp , typename _Compare >
__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_comp [protected]
__Compare to use.
```

`_M_first_insert`

```
template<typename _Tp , typename _Compare >
bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_first_insert [protected]
```

State flag that determines whether the `_LoserTree` is empty.

Only used for building the `_LoserTree`.

Referenced by [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#), and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#).

`_M_log_k`

```
template<typename _Tp , typename _Compare >
unsigned int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_log_k [protected]
```

`log_2[_M_k]`

Referenced by [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#).

`_M_losers`

```
template<typename _Tp , typename _Compare >
__Loser* __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_losers [protected]
```

`_LoserTree` elements.

Referenced by [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#), [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#), and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#).

[__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_get_min_source\(\)](#), and [__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase\(\)](#).

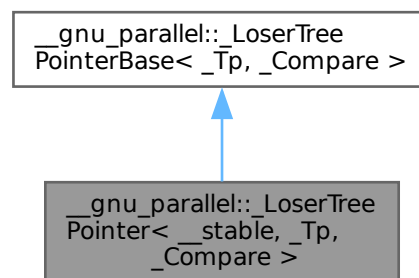
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.117 `__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >`:



Public Member Functions

- **_LoserTreePointer** (unsigned int __k, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (const _Tp &__key, bool __sup)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- unsigned int **_M_ik**
- unsigned int **_M_offset**

6.117.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >
```

Stable _LoserTree implementation.

The unstable variant is implemented using partial instantiation below.

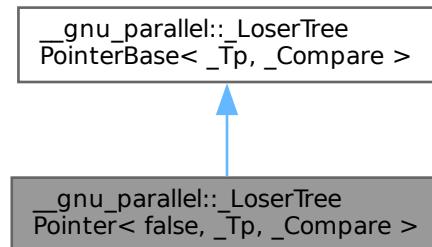
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.118 __gnu_parallel::_LoserTreePointer< false, _Tp, _Compare > Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for __gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >:



Public Member Functions

- **_LoserTreePointer** (unsigned int __k, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (const _Tp &__key, bool __sup)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_offset`

6.118.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >
```

Unstable `_LoserTree` implementation.

The stable variant is above.

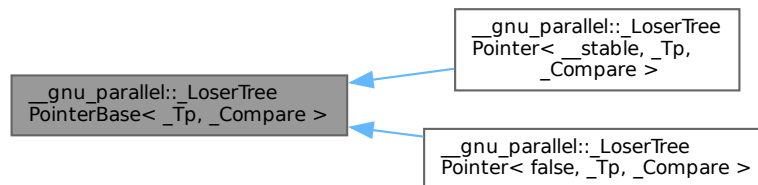
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.119 `__gnu_parallel::_LoserTreePointerBase<_Tp, _Compare>` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreePointerBase<_Tp, _Compare>`:

**Classes**

- struct [_Loser](#)

Public Member Functions

- `_LoserTreePointerBase` (unsigned int `__k`, `_Compare` `__comp=std::less<_Tp>()`)
- int `__get_min_source` ()
- void `__insert_start` (const `_Tp` &`__key`, int `__source`, bool `__sup`)

Protected Attributes

- `_Compare` `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- [_Loser](#) * `_M_losers`
- unsigned int `_M_offset`

6.119.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >
```

Base class of _Loser Tree implementation using pointers.

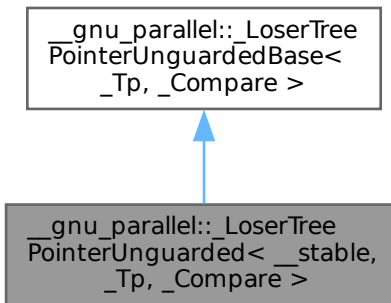
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.120 __gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare > Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for __gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >:



Public Member Functions

- `_LoserTreePointerUnguarded` (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void `__delete_min_insert` (const _Tp &__key, bool __sup)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int __root)
- void `__insert_start` (const _Tp &__key, int __source, bool)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_offset`

6.120.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >
```

Stable unguarded _LoserTree variant storing pointers.

Unstable variant is implemented below using partial specialization.

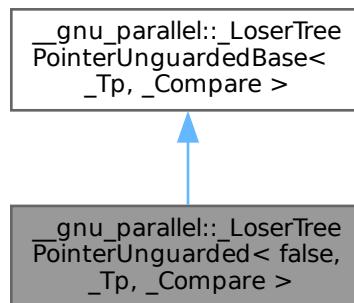
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.121 `__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >`:



Public Member Functions

- `_LoserTreePointerUnguarded` (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void `__delete_min_insert` (const _Tp &__key, bool __sup)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int __root)
- void `__insert_start` (const _Tp &__key, int __source, bool)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_offset`

6.121.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >
```

Unstable unguarded `_LoserTree` variant storing pointers.

Stable variant is above.

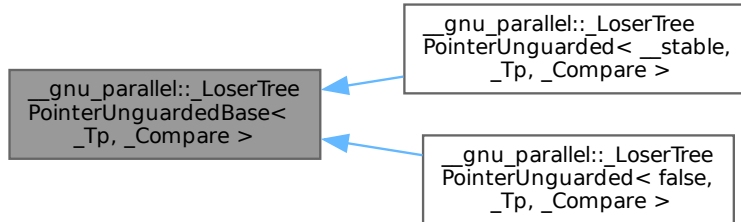
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.122 `__gnu_parallel::_LoserTreePointerUnguardedBase<_Tp, _Compare>` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreePointerUnguardedBase<_Tp, _Compare>`:



Public Member Functions

- `_LoserTreePointerUnguardedBase` (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less<_Tp>](#)())
- int `__get_min_source` ()
- void `__insert_start` (const _Tp &__key, int __source, bool)

Protected Attributes

- _Compare `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- _Loser * `_M_losers`
- unsigned int `_M_offset`

6.122.1 Detailed Description

```
template<typename _Tp, typename _Compare>
```

```
class __gnu_parallel::_LoserTreePointerUnguardedBase<_Tp, _Compare>
```

Unguarded loser tree, keeping only pointers to the elements in the tree structure.

No guarding is done, therefore not a single input sequence must run empty. This is a very fast variant.

The documentation for this class was generated from the following file:

- [losertree.h](#)

6.123 `__gnu_parallel::_LoserTreeTraits<_Tp>` Struct Template Reference

```
#include <multiway_merge.h>
```

Static Public Attributes

- static const bool `_M_use_pointer`

6.123.1 Detailed Description

```
template<typename _Tp>
struct __gnu_parallel::_LoserTreeTraits< _Tp >
```

Traits for determining whether the loser tree should use pointers or copies.

The field “`_M_use_pointer`” is used to determine whether to use pointers in the loser trees or whether to copy the values into the loser tree.

The default behavior is to use pointers if the data type is 4 times as big as the pointer to it.

Specialize for your data type to customize the behavior.

Example:

```
template<> struct _LoserTreeTraits<int> { static const bool _M_use_pointer = false; };
template<> struct _LoserTreeTraits<heavyweight_type> { static const bool _M_use_pointer = true; };
```

Parameters

<code>_Tp</code>	type to give the loser tree traits for.
------------------	---

6.123.2 Member Data Documentation

`_M_use_pointer`

```
template<typename _Tp >
const bool __gnu_parallel::_LoserTreeTraits< _Tp >::_M_use_pointer [static]
```

True iff to use pointers instead of values in loser trees.

The default behavior is to use pointers if the data type is four times as big as the pointer to it.

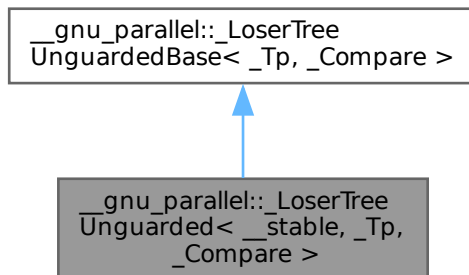
The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.124 `__gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >`:



Public Member Functions

- **_LoserTreeUnguarded** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (_Tp __key, bool)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool)

Protected Attributes

- unsigned int **_M_ik**
- unsigned int **_M_offset**

6.124.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >
```

Stable implementation of unguarded _LoserTree.

Unstable variant is selected below with partial specialization.

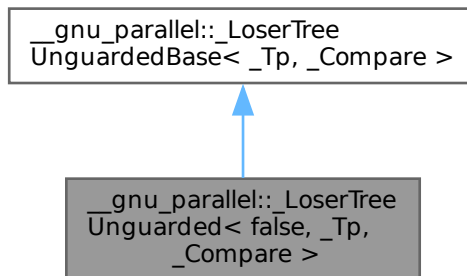
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.125 __gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare > Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for __gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >:



Public Member Functions

- **_LoserTreeUnguarded** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (_Tp __key, bool)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool)

Protected Attributes

- unsigned int `_M_ik`
- unsigned int `_M_offset`

6.125.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >
```

Non-Stable implementation of unguarded `_LoserTree`.

Stable implementation is above.

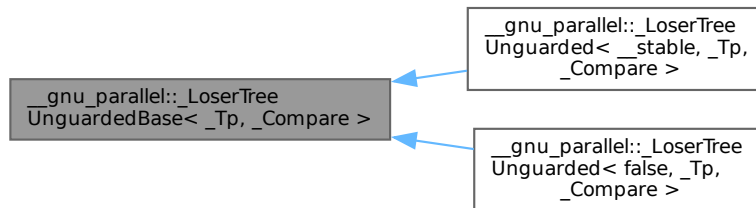
The documentation for this class was generated from the following file:

- [losertree.h](#)

6.126 `__gnu_parallel::_LoserTreeUnguardedBase<_Tp, _Compare>` Class Template Reference

```
#include <losertree.h>
```

Inheritance diagram for `__gnu_parallel::_LoserTreeUnguardedBase<_Tp, _Compare>`:

**Public Member Functions**

- `_LoserTreeUnguardedBase` (unsigned int `__k`, const `_Tp` & `__sentinel`, `_Compare` `__comp=std::less<_Tp>()`)
- int `__get_min_source` ()
- void `__insert_start` (const `_Tp` & `__key`, int `__source`, bool)

Protected Attributes

- `_Compare` `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- `_Loser` * `_M_losers`
- unsigned int `_M_offset`

6.126.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguardedBase<_Tp, _Compare>
```

Base class for unguarded `_LoserTree` implementation.

The whole element is copied into the tree structure.

No guarding is done, therefore not a single input sequence must run empty. Unused `__sequence` heads are marked with a sentinel which is `>` all elements that are to be merged.

This is a very fast variant.

The documentation for this class was generated from the following file:

- [losertree.h](#)

6.127 `__gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >` Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `detail::rebind_traits< _Alloc, Metadata >::const_reference` **const_reference**
- typedef `Metadata` **metadata_type**

Public Member Functions

- `const_reference` **get_metadata** () const

Public Attributes

- `metadata_type` **m_metadata**

6.127.1 Detailed Description

```
template<typename Metadata, typename _Alloc>
struct __gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >
```

Metadata base primary template.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.128 `__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >` Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `null_type` **metadata_type**

6.128.1 Detailed Description

```
template<typename _Alloc>
struct __gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >
```

Specialization for null metadata.

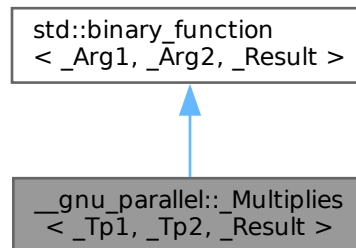
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.129 `__gnu_parallel::_Multiplies<_Tp1, _Tp2, _Result>` Struct Template Reference

```
#include <base.h>
```

Inheritance diagram for `__gnu_parallel::_Multiplies<_Tp1, _Tp2, _Result>`:

**Public Types**

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Result` **operator()** (const `_Tp1` &`__x`, const `_Tp2` &`__y`) const

6.129.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Result = __typeof__((*static_cast<_Tp1*>(0) * *static_cast<_Tp2*>(0)))>
struct __gnu_parallel::_Multiplies<_Tp1, _Tp2, _Result>
```

Similar to `std::multiplies`, but allows two different types.

6.129.2 Member Typedef Documentation**first_argument_type**

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg1 std::binary\_function<\_Arg1, \_Arg2, \_Result>::first\_argument\_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Result std::binary\_function<\_Arg1, \_Arg2, \_Result>::result\_type [inherited]
result_type is the return type
```

second_argument_type

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg2 std::binary\_function< _Arg1, _Arg2, _Result >::second_argument_type [inherited]
```

`second_argument_type` is the type of the second argument

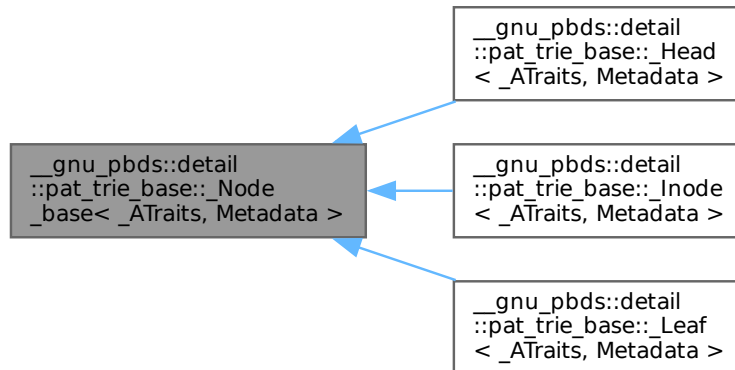
The documentation for this struct was generated from the following file:

- [base.h](#)

6.130 `__gnu_pbds::detail::pat_trie_base::_Node_base<_ATraits, Metadata >` Struct Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Node_base<_ATraits, Metadata >`:



Public Types

- typedef `_ATraits::const_iterator` **a_const_iterator**
- typedef `detail::rebind_traits<_Alloc, _ATraits>::const_pointer` **a_const_pointer**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `detail::rebind_traits<_Alloc, _Node_base>::pointer` **node_pointer**
- typedef `_ATraits::type_traits` **type_traits**

Public Member Functions

- `_Node_base (node_type type)`

Public Attributes

- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

6.130.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >
```

Node base.

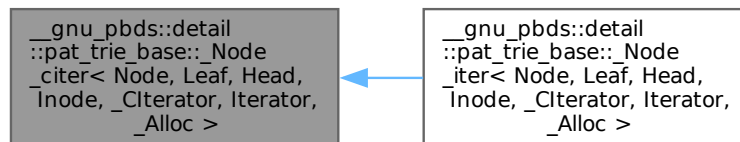
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.131 `__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Citerator, Iterator, _Alloc >` Class Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Citerator, Iterator, _Alloc >`:



Public Types

- typedef value_type **const_reference**
- typedef [trivial_iterator_difference_type](#) **difference_type**
- typedef [trivial_iterator_tag](#) **iterator_category**
- typedef [rebind_traits](#)< _Alloc, [metadata_type](#) >::const_reference **metadata_const_reference**
- typedef Node::metadata_type **metadata_type**
- typedef value_type **reference**
- typedef _Alloc::size_type **size_type**
- typedef _Citerator **value_type**

Public Member Functions

- **_Node_citer** (node_pointer p_nd=0, a_const_pointer p_traits=0)
- [_Node_citer get_child](#) (size_type i) const
- [metadata_const_reference get_metadata](#) () const
- size_type [num_children](#) () const
- bool [operator!=](#) (const [_Node_citer](#) &other) const
- const_reference [operator*](#) () const
- bool [operator==](#) (const [_Node_citer](#) &other) const
- [std::pair](#)< a_const_iterator, a_const_iterator > [valid_prefix](#) () const

Public Attributes

- node_pointer **m_p_nd**
- a_const_pointer **m_p_traits**

Protected Types

- typedef Node::a_const_iterator **a_const_iterator**
- typedef Node::a_const_pointer **a_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Inode >::const_pointer **inode_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Inode >::pointer **inode_pointer**
- typedef [rebind_traits](#)< _Alloc, Leaf >::const_pointer **leaf_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Leaf >::pointer **leaf_pointer**
- typedef [rebind_traits](#)< _Alloc, Node >::pointer **node_pointer**

6.131.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, typename _CIterator, typename
Iterator, typename _Alloc>
class __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >
```

Node const iterator.

6.131.2 Member Typedef Documentation

metadata_const_reference

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
typedef rebind\_traits<_Alloc,metadata\_type>::const_reference \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer<
Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::metadata_const_reference
Const metadata reference type.
```

metadata_type

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
typedef Node::metadata_type \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head,
Inode, _CIterator, Iterator, _Alloc >::metadata_type
Metadata type.
```

6.131.3 Member Function Documentation

get_child()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
\_Node\_citer \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::get_child (
    size_type i ) const [inline]
Returns a __const node __iterator to the corresponding node's i-th child.
References std::advance\(\).
```

get_metadata()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
metadata\_const\_reference \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head, Inode,
_CIterator, Iterator, _Alloc >::get_metadata ( ) const [inline]
Metadata access.
```

num_children()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
size_type __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::num_children ( ) const [inline]
```

Returns the number of children in the corresponding node.

References [std::distance\(\)](#).

Referenced by [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::operator*\(\)](#), and [__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::operator*\(\)](#).

operator"!=()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator,
_Alloc >::operator!= (
    const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other )
const [inline]
```

Compares content (negatively) to a different iterator object.

operator*()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
const_reference __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _C
Iterator, Iterator, _Alloc >::operator* ( ) const [inline]
```

Const access; returns the __const iterator* associated with the current leaf.

References [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::num_children\(\)](#).

operator==()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator,
_Alloc >::operator== (
    const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other )
const [inline]
```

Compares content to a different iterator object.

valid_prefix()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
std::pair< a_const_iterator, a_const_iterator > __gnu_pbds::detail::pat_trie_base::_Node_citer<
Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::valid_prefix ( ) const [inline]
```

Subtree valid prefix.

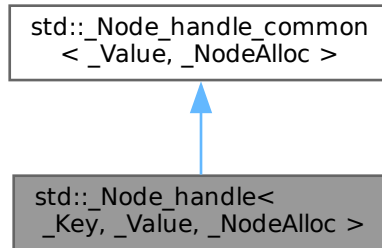
The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

6.132 std::_Node_handle<_Key,_Value,_NodeAlloc > Class Template Reference

```
#include <node_handle.h>
```


Inheritance diagram for `std::_Node_handle< _Key, _Value, _NodeAlloc >`:



Public Types

- using **allocator_type** = `__alloc_rebind< _NodeAlloc, _Value >`
- using **key_type** = `_Key`
- using **mapped_type** = `typename _Value::second_type`

Public Member Functions

- **_Node_handle** ([_Node_handle](#) &&) noexcept=default
- bool **empty** () const noexcept
- allocator_type **get_allocator** () const noexcept
- key_type & **key** () const noexcept
- mapped_type & **mapped** () const noexcept
- **operator bool** () const noexcept
- [_Node_handle](#) & **operator=** ([_Node_handle](#) &&) noexcept=default
- void **swap** ([_Node_handle](#) &__nh) noexcept

Friends

- template<typename _Key2, typename _Value2, typename _ValueAlloc, typename _ExtractKey, typename _Equal, typename _Hash, typename _RangeHash, typename _Unused, typename _RehashPolicy, typename _Traits >
class **_Hashtable**
- template<typename _Key2, typename _Value2, typename _KeyOfValue, typename _Compare, typename _ValueAlloc >
class **_Rb_tree**
- void **swap** ([_Node_handle](#) &__x, [_Node_handle](#) &__y) noexcept(noexcept(__x.swap(__y)))

6.132.1 Detailed Description

```
template<typename _Key, typename _Value, typename _NodeAlloc>
class std::_Node_handle< _Key, _Value, _NodeAlloc >
```

Node handle type for maps.

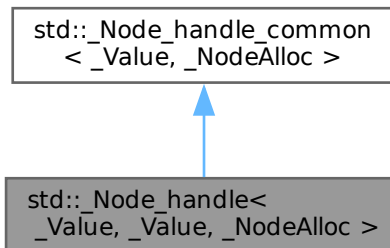
The documentation for this class was generated from the following file:

- [node_handle.h](#)

6.133 std::_Node_handle< _Value, _Value, _NodeAlloc > Class Template Reference

```
#include <node_handle.h>
```

Inheritance diagram for std::_Node_handle< _Value, _Value, _NodeAlloc >:

**Public Types**

- using **allocator_type** = __alloc_rebind< _NodeAlloc, _Value >
- using **value_type** = _Value

Public Member Functions

- **_Node_handle** ([_Node_handle](#) &&) noexcept=default
- bool **empty** () const noexcept
- allocator_type **get_allocator** () const noexcept
- **operator bool** () const noexcept
- [_Node_handle](#) & **operator=** ([_Node_handle](#) &&) noexcept=default
- void **swap** ([_Node_handle](#) &__nh) noexcept
- value_type & **value** () const noexcept

Friends

- template<typename _Key2, typename _Value2, typename _ValueAlloc, typename _ExtractKey, typename _Equal, typename _Hash, typename _RangeHash, typename _Unused, typename _RehashPolicy, typename _Traits >
class **_Hashtable**
- template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >
class **_Rb_tree**
- void **swap** ([_Node_handle](#) &__x, [_Node_handle](#) &__y) noexcept(noexcept(__x.swap(__y)))

6.133.1 Detailed Description

```
template<typename _Value, typename _NodeAlloc>
class std::_Node_handle< _Value, _Value, _NodeAlloc >
```

Node handle type for sets.

The documentation for this class was generated from the following file:

- [node_handle.h](#)

6.134 `std::_Node_handle_common<_Val, _NodeAlloc >` Class Template Reference

```
#include <node_handle.h>
```

Public Types

- using **allocator_type** = `__alloc_rebind<_NodeAlloc, _Val >`

Public Member Functions

- `bool empty ()` const noexcept
- `allocator_type get_allocator ()` const noexcept
- `operator bool ()` const noexcept

6.134.1 Detailed Description

```
template<typename _Val, typename _NodeAlloc>
class std::_Node_handle_common<_Val, _NodeAlloc >
```

Base class for node handle types of maps and sets.

The documentation for this class was generated from the following file:

- [node_handle.h](#)

6.135 `std::_Node_insert_return<_Iterator, _NodeHandle >` Struct Template Reference

```
#include <node_handle.h>
```

Public Attributes

- `bool inserted`
- `_NodeHandle node`
- `_Iterator position`

6.135.1 Detailed Description

```
template<typename _Iterator, typename _NodeHandle>
struct std::_Node_insert_return<_Iterator, _NodeHandle >
```

Return type of `insert(node_handle&&)` on unique maps/sets.

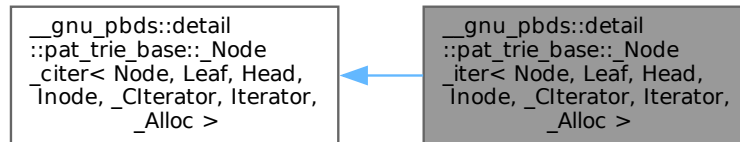
The documentation for this struct was generated from the following file:

- [node_handle.h](#)

6.136 `__gnu_pbds::detail::pat_trie_base::_Node_iter<Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >` Class Template Reference

```
#include <pat_trie_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >`:



Public Types

- typedef value_type **const_reference**
- typedef [trivial_iterator_difference_type](#) **difference_type**
- typedef [trivial_iterator_tag](#) **iterator_category**
- typedef [rebind_traits](#)< _Alloc, [metadata_type](#) >::const_reference [metadata_const_reference](#)
- typedef Node::metadata_type [metadata_type](#)
- typedef value_type **reference**
- typedef base_type::size_type **size_type**
- typedef Iterator **value_type**

Public Member Functions

- **_Node_iter** (node_pointer p_nd=0, a_const_pointer p_traits=0)
- **_Node_iter** [get_child](#) (size_type i) const
- [metadata_const_reference](#) [get_metadata](#) () const
- size_type [num_children](#) () const
- bool [operator!=](#) (const [_Node_citer](#) &other) const
- reference [operator*](#) () const
- bool [operator==](#) (const [_Node_citer](#) &other) const
- [std::pair](#)< a_const_iterator, a_const_iterator > [valid_prefix](#) () const

Public Attributes

- node_pointer **m_p_nd**
- a_const_pointer **m_p_traits**

Protected Types

- typedef Node::a_const_iterator **a_const_iterator**
- typedef [rebind_traits](#)< _Alloc, Inode >::const_pointer **inode_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Leaf >::const_pointer **leaf_const_pointer**
- typedef [rebind_traits](#)< _Alloc, Leaf >::pointer **leaf_pointer**

6.136.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, typename _CIterator, typename
Iterator, typename _Alloc>
```

```
class __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >
```

Node iterator.

6.136.2 Member Typedef Documentation

metadata_const_reference

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
typedef rebind\_traits<_Alloc,metadata_type>::const_reference \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer<
Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::metadata_const_reference [inherited]
Const metadata reference type.
```

metadata_type

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
typedef Node::metadata_type \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head,
Inode, _CIterator, Iterator, _Alloc >::metadata_type [inherited]
Metadata type.
```

6.136.3 Member Function Documentation

get_child()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
\_Node\_iter \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_iter< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::get_child (
    size_type i ) const [inline]
```

Returns a node __iterator to the corresponding node's i-th child.

References [std::advance\(\)](#).

get_metadata()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
metadata\_const\_reference \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head, Inode,
_CIterator, Iterator, _Alloc >::get_metadata ( ) const [inline], [inherited]
```

Metadata access.

num_children()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
size_type \_\_gnu\_pbds::detail::pat\_trie\_base::\_Node\_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::num_children ( ) const [inline], [inherited]
```

Returns the number of children in the corresponding node.

References [std::distance\(\)](#).

Referenced by [__gnu_pbds::detail::pat_trie_base::_Node_citer](#)< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::operator*(), and [__gnu_pbds::detail::pat_trie_base::_Node_iter](#)< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::operator*().

operator"!=()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator,
_Alloc >::operator!= (
    const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other )
const [inline], [inherited]
```

Compares content (negatively) to a different iterator object.

operator*()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
reference __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::operator* ( ) const [inline]
```

Access; returns the iterator* associated with the current leaf.

References `__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::num_children()`.

operator==(())

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator,
_Alloc >::operator==(
    const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other )
const [inline], [inherited]
```

Compares content to a different iterator object.

valid_prefix()

```
template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator ,
typename Iterator , typename _Alloc >
std::pair< a_const_iterator, a_const_iterator > __gnu_pbds::detail::pat_trie_base::_Node_citer<
Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::valid_prefix ( ) const [inline], [inherited]
```

Subtree valid prefix.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

6.137 `__gnu_debug::_Not_equal_to<_Type>` Class Template Reference

```
#include <safe_sequence.h>
```

Public Member Functions

- `_Not_equal_to` (const `_Type` &__v)
- bool `operator()` (const `_Type` &__x) const

6.137.1 Detailed Description

```
template<typename _Type>
class __gnu_debug::_Not_equal_to<_Type>
```

A simple function object that returns true if the passed-in value is not equal to the stored value. It saves typing over using both `bind1st` and `not_equal`.

The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

6.138 std::_Not_fn<_Fn> Class Template Reference

Public Member Functions

- `template<typename _Fn2 >`
`constexpr _Not_fn (_Fn2 && __fn, int)`
- `_Not_fn (_Not_fn && __fn)=default`
- `_Not_fn (const _Not_fn & __fn)=default`

Public Attributes

- `template<typename... _Args>`
`constexpr decltype(_S_not< __inv_res_t< _Fn &&, _Args... > >()) operator() (_Args &&... __args)`
`&&noexcept(_is_nothrow_invocable< _Fn &&, _Args... >::value &&noexcept(_S_not< __inv_res_t< _Fn`
`&&, _Args... > >()))`
- `template<typename... _Args>`
`constexpr decltype(_S_not< __inv_res_t< _Fn &, _Args... > >()) operator() (_Args &&... __args) &noexcept(↵`
`_is_nothrow_invocable< _Fn &, _Args... >::value &&noexcept(_S_not< __inv_res_t< _Fn &, _Args... > >()))`
- `template<typename... _Args>`
`constexpr decltype(_S_not< __inv_res_t< _Fn const &&, _Args... > >()) operator() (_Args &&... __args) const`
`&&noexcept(_is_nothrow_invocable< _Fn const &&, _Args... >::value &&noexcept(_S_not< __inv_res_t< _Fn`
`const &&, _Args... > >()))`
- `template<typename... _Args>`
`constexpr decltype(_S_not< __inv_res_t< _Fn const &, _Args... > >()) operator() (_Args &&... __args) const`
`&noexcept(_is_nothrow_invocable< _Fn const &, _Args... >::value &&noexcept(_S_not< __inv_res_t< _Fn`
`const &, _Args... > >()))`

6.138.1 Detailed Description

`template<typename _Fn>`
`class std::_Not_fn<_Fn>`

Generalized negator.

The documentation for this class was generated from the following file:

- [functional](#)

6.139 __gnu_parallel::_Nothing Struct Reference

```
#include <for_each_selectors.h>
```

Public Member Functions

- `template<typename _It >`
`void operator() (_It __i)`

6.139.1 Detailed Description

Functor doing nothing.

For some __reduction tasks (this is not a function object, but is passed as __selector __dummy parameter.

6.139.2 Member Function Documentation

operator>()()

```
template<typename _It >
void __gnu_parallel::_Nothing::operator() (
    _It __i ) [inline]
```

Functor execution.

Parameters

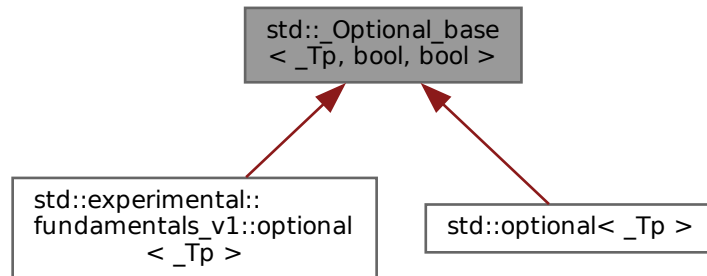
↩	iterator referencing object.
↩	
↩	
↩	
<i>i</i>	

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

6.140 std::_Optional_base< _Tp, bool, bool > Struct Template Reference

Inheritance diagram for std::_Optional_base< _Tp, bool, bool >:



Public Member Functions

- constexpr **_Optional_base** ([_Optional_base](#) &&__other) noexcept(is_nothrow_move_constructible_v< _Tp >)
- constexpr **_Optional_base** (const [_Optional_base](#) &__other)
- template<typename... _Args, [enable_if_t](#)< is_constructible_v< _Tp, _Args... >, bool > = false>
constexpr **_Optional_base** (in_place_t, _Args &&... __args)
- template<typename _Up, typename... _Args, [enable_if_t](#)< is_constructible_v< _Tp, [initializer_list](#)< _Up > &, _Args... >, bool > = false>
constexpr **_Optional_base** (in_place_t, [initializer_list](#)< _Up > __il, _Args &&... __args)
- [_Optional_base](#) & **operator=** ([_Optional_base](#) &&)=default
- [_Optional_base](#) & **operator=** (const [_Optional_base](#) &)=default

Public Attributes

- [_Optional_payload](#)< _Tp > **_M_payload**

Protected Types

- using **_Stored_type** = [remove_const_t](#)< _Tp >

Protected Member Functions

- `template<typename... _Args>`
`constexpr void _M_construct (_Args &&... __args) noexcept(is_nothrow_constructible_v< _Stored_type, _Args... >)`
- `constexpr void _M_destruct () noexcept`
- `constexpr const _Tp & _M_get () const noexcept`
- `constexpr _Tp & _M_get () noexcept`
- `constexpr bool _M_is_engaged () const noexcept`
- `constexpr void _M_reset () noexcept`

6.140.1 Detailed Description

```
template<typename _Tp, bool = is_trivially_copy_constructible_v<_Tp>, bool = is_trivially_move_constructible_v<_Tp>>
struct std::Optional_base< _Tp, bool, bool >
```

Class template that provides copy/move constructors of optional.

Such a separate base class template is necessary in order to conditionally make copy/move constructors trivial.

When the contained value is trivially copy/move constructible, the copy/move constructors of `_Optional_base` will invoke the trivial copy/move constructor of `_Optional_payload`. Otherwise, they will invoke `_Optional_payload(bool, const _Optional_payload&)` or `_Optional_payload(bool, _Optional_payload&&)` to initialize the contained value, if copying/moving an engaged optional.

Whether the other special members are trivial is determined by the `_Optional_payload<_Tp>` specialization used for the `_M_payload` member.

See also

`optional`, `_Enable_special_members`

The documentation for this struct was generated from the following file:

- [optional](#)

6.141 `__gnu_parallel::Piece<_DifferenceTp>` Struct Template Reference

```
#include <multiway_mergesort.h>
```

Public Types

- `typedef _DifferenceTp _DifferenceType`

Public Attributes

- `_DifferenceType _M_begin`
- `_DifferenceType _M_end`

6.141.1 Detailed Description

```
template<typename _DifferenceTp>
struct __gnu_parallel::Piece<_DifferenceTp >
```

Subsequence description.

6.141.2 Member Data Documentation

_M_begin

```
template<typename _DifferenceTp >
_DifferenceType __gnu_parallel::_Piece< _DifferenceTp >::_M_begin
Begin of subsequence.
```

_M_end

```
template<typename _DifferenceTp >
_DifferenceType __gnu_parallel::_Piece< _DifferenceTp >::_M_end
End of subsequence.
```

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

6.142 std::_Placeholder<_Num> Struct Template Reference

6.142.1 Detailed Description

```
template<int _Num>
struct std::_Placeholder<_Num>
```

The type of placeholder objects defined by libstdc++.

Since

C++11

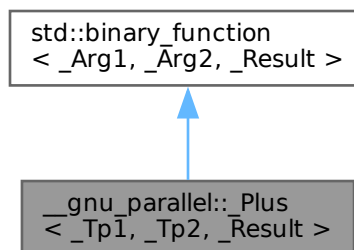
The documentation for this struct was generated from the following file:

- [functional](#)

6.143 __gnu_parallel::_Plus<_Tp1, _Tp2, _Result> Struct Template Reference

```
#include <base.h>
```

Inheritance diagram for __gnu_parallel::_Plus<_Tp1, _Tp2, _Result>:



Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Result` **operator()** (const `_Tp1` &__x, const `_Tp2` &__y) const

6.143.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Result = __typeof__(*static_cast<_Tp1*>(0) + *static_cast<_Tp2*>(0))>
struct __gnu_parallel::__Plus<_Tp1, _Tp2, _Result >
```

Similar to `std::plus`, but allows two different types.

6.143.2 Member Typedef Documentation

`first_argument_type`

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg1 std::binary\_function< _Arg1, _Arg2, _Result >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

`result_type`

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Result std::binary\_function< _Arg1, _Arg2, _Result >::result_type [inherited]
result_type is the return type
```

`second_argument_type`

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg2 std::binary\_function< _Arg1, _Arg2, _Result >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [base.h](#)

6.144 `__gnu_parallel::__PMWSSortingData<_RAIter >` Struct Template Reference

```
#include <multiway_mergesort.h>
```

Public Types

- typedef `_TraitsType::difference_type` `_DifferenceType`
- typedef `std::iterator_traits<_RAIter >` `_TraitsType`
- typedef `_TraitsType::value_type` `_ValueType`

Public Attributes

- `_ThreadIndex` `_M_num_threads`
- `_DifferenceType` * `_M_offsets`
- `std::vector<_Piece<_DifferenceType > >` * `_M_pieces`

- `_ValueType * _M_samples`
- `_RAIter _M_source`
- `_DifferenceType * _M_starts`
- `_ValueType ** _M_temporary`

6.144.1 Detailed Description

`template<typename _RAIter>`
`struct __gnu_parallel::_PMWMSortingData<_RAIter>`

Data accessed by all threads.

PMWMS = parallel multiway mergesort

6.144.2 Member Data Documentation

`_M_num_threads`

`template<typename _RAIter>`
`_ThreadIndex __gnu_parallel::_PMWMSortingData<_RAIter>::_M_num_threads`

Number of threads involved.

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

`_M_offsets`

`template<typename _RAIter>`
`_DifferenceType* __gnu_parallel::_PMWMSortingData<_RAIter>::_M_offsets`

Offsets to add to the found positions.

Referenced by `__gnu_parallel::parallel_sort_mwms()`.

`_M_pieces`

`template<typename _RAIter>`
`std::vector<_Piece<_DifferenceType>>* __gnu_parallel::_PMWMSortingData<_RAIter>::_M_pieces`

Pieces of data to merge [thread][__sequence].

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

`_M_samples`

`template<typename _RAIter>`
`_ValueType* __gnu_parallel::_PMWMSortingData<_RAIter>::_M_samples`

Samples.

Referenced by `__gnu_parallel::__determine_samples()`, and `__gnu_parallel::parallel_sort_mwms()`.

`_M_source`

`template<typename _RAIter>`
`_RAIter __gnu_parallel::_PMWMSortingData<_RAIter>::_M_source`

Input __begin.

Referenced by `__gnu_parallel::__determine_samples()`, `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

`_M_starts`

`template<typename _RAIter>`
`_DifferenceType* __gnu_parallel::_PMWMSortingData<_RAIter>::_M_starts`

Start indices, per thread.

Referenced by `__gnu_parallel::__determine_samples()`, `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

`_M_temporary`

```
template<typename _RAIter >
```

```
_ValueType** \_\_gnu\_parallel::\_PMWMSortingData< _RAIter >::_M_temporary
```

Storage in which to sort.

Referenced by [__gnu_parallel::parallel_sort_mwms\(\)](#), and [__gnu_parallel::parallel_sort_mwms_pu\(\)](#).

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

6.145 `__gnu_cxx::Pointer_adapter<_Storage_policy>` Class Template Reference

```
#include <pointer.h>
```

Inherits `_Storage_policy`.

Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef `_Storage_policy::element_type` **element_type**
- typedef `std::random_access_iterator_tag` **iterator_category**
- typedef `_Pointer_adapter` **pointer**
- typedef `_Reference_type< element_type >::reference` **reference**
- typedef `_Unqualified_type< element_type >::type` **value_type**

Public Member Functions

- `template<typename _Up >`
`_Pointer_adapter` (`_Up *``__arg`)
- `_Pointer_adapter` (`const` `_Pointer_adapter` &`__arg`)
- `template<typename _Up >`
`_Pointer_adapter` (`const` `_Pointer_adapter`< `_Up` > &`__arg`)
- `_Pointer_adapter` (`element_type *``__arg`=0)
- `operator bool` () `const`
- `reference operator*` () `const`
- `_Pointer_adapter` & `operator++` ()
- `_Pointer_adapter` `operator++` (`int`)
- `_Pointer_adapter` & `operator+=` (`int` `__offset`)
- `_Pointer_adapter` & `operator+=` (`long` `__offset`)
- `_Pointer_adapter` & `operator+=` (`long long` `__offset`)
- `_Pointer_adapter` & `operator+=` (`short` `__offset`)
- `_Pointer_adapter` & `operator+=` (`unsigned int` `__offset`)
- `_Pointer_adapter` & `operator+=` (`unsigned long` `__offset`)
- `_Pointer_adapter` & `operator+=` (`unsigned long long` `__offset`)
- `_Pointer_adapter` & `operator+=` (`unsigned short` `__offset`)
- `template<typename _Up >`
`std::ptrdiff_t operator-` (`const` `_Pointer_adapter`< `_Up` > &`__rhs`) `const`
- `_Pointer_adapter` & `operator--` ()
- `_Pointer_adapter` `operator--` (`int`)
- `_Pointer_adapter` & `operator-=` (`int` `__offset`)
- `_Pointer_adapter` & `operator-=` (`long` `__offset`)
- `_Pointer_adapter` & `operator-=` (`long long` `__offset`)
- `_Pointer_adapter` & `operator-=` (`short` `__offset`)
- `_Pointer_adapter` & `operator-=` (`unsigned int` `__offset`)

- `Pointer_adapter` & **operator=** (unsigned long __offset)
- `Pointer_adapter` & **operator=** (unsigned long long __offset)
- `Pointer_adapter` & **operator=** (unsigned short __offset)
- `element_type * operator->` () const
- `template<typename _Up >`
`Pointer_adapter` & **operator=** (_Up *__arg)
- `Pointer_adapter` & **operator=** (const `Pointer_adapter` & __arg)
- `template<typename _Up >`
`Pointer_adapter` & **operator=** (const `Pointer_adapter`< _Up > & __arg)
- `reference operator[]` (std::ptrdiff_t __index) const

Friends

- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, int __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, long __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, long long __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, short __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, unsigned int __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, unsigned long __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, unsigned long long __offset)
- `Pointer_adapter` **operator+** (const `Pointer_adapter` & __lhs, unsigned short __offset)
- `Pointer_adapter` **operator+** (int __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (long __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (long long __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (short __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (unsigned int __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (unsigned long __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (unsigned long long __offset, const `Pointer_adapter` & __rhs)
- `Pointer_adapter` **operator+** (unsigned short __offset, const `Pointer_adapter` & __rhs)
- `template<typename _Up >`
`std::ptrdiff_t operator-` (_Up *__lhs, const `Pointer_adapter` & __rhs)
- `template<typename _Up >`
`std::ptrdiff_t operator-` (const `Pointer_adapter` & __lhs, _Up *__rhs)
- `std::ptrdiff_t operator-` (const `Pointer_adapter` & __lhs, `element_type` *__rhs)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, int __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, long __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, long long __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, short __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, unsigned int __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, unsigned long __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, unsigned long long __offset)
- `Pointer_adapter` **operator-** (const `Pointer_adapter` & __lhs, unsigned short __offset)
- `std::ptrdiff_t operator-` (`element_type` *__lhs, const `Pointer_adapter` & __rhs)

6.145.1 Detailed Description

```
template<typename _Storage_policy>
class __gnu_cxx::_Pointer_adapter< _Storage_policy >
```

The following provides an 'alternative pointer' that works with the containers when specified as the pointer typedef of the allocator.

The pointer type used with the containers doesn't have to be this class, but it must support the implicit conversions, pointer arithmetic, comparison operators, etc. that are supported by this class, and avoid raising compile-time ambiguities. Because creating a working pointer can be challenging, this pointer template was designed to wrapper an easier storage policy type, so that it becomes reusable for creating other pointer types.

A key point of this class is also that it allows container writers to 'assume' `Allocator::pointer` is a typedef for a normal pointer. This class supports most of the conventions of a true pointer, and can, for instance handle implicit conversion to const and base class pointer types. The only impositions on container writers to support extended pointers are: 1) use the `Allocator::pointer` typedef appropriately for pointer types. 2) if you need pointer casting, use the `__pointer_cast<>` functions from `ext/cast.h`. This allows pointer cast operations to be overloaded as necessary by custom pointers.

Note: The const qualifier works with this pointer adapter as follows:

```
_Tp* == _Pointer_adapter<_Std_pointer_impl<_Tp> >; const _Tp* == _Pointer_adapter<_Std_pointer_impl<const
_Tp> >; _Tp* const == const _Pointer_adapter<_Std_pointer_impl<_Tp> >; const _Tp* const == const _Pointer_
adapter<_Std_pointer_impl<const _Tp> >;
```

The documentation for this class was generated from the following file:

- [pointer.h](#)

6.146 __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp > Class Template Reference

```
#include <base.h>
```

Public Types

- typedef `_DifferenceTp` **DifferenceType**
- typedef `_PseudoSequenceIterator< _Tp, uint64_t >` **iterator**

Public Member Functions

- `_PseudoSequence` (const `_Tp` &__val, `DifferenceType` __count)
- `iterator begin` () const
- `iterator end` () const

6.146.1 Detailed Description

```
template<typename _Tp, typename _DifferenceTp>
class __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >
```

Sequence that conceptually consists of multiple copies of the same element. The copies are not stored explicitly, of course.

Parameters

<code>_Tp</code>	Sequence <code>_M_value</code> type.
<code>_DifferenceTp</code>	Sequence difference type.

6.146.2 Constructor & Destructor Documentation

`__PseudoSequence()`

```
template<typename _Tp , typename _DifferenceTp >
__gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >::_PseudoSequence (
    const _Tp & __val,
    _DifferenceType __count ) [inline]
```

Constructor.

Parameters

<code>__val</code>	Element of the sequence.
<code>__count</code>	Number of (virtual) copies.

6.146.3 Member Function Documentation

`begin()`

```
template<typename _Tp , typename _DifferenceTp >
iterator __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >::begin ( ) const [inline]
```

Begin iterator.

`end()`

```
template<typename _Tp , typename _DifferenceTp >
iterator __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >::end ( ) const [inline]
```

End iterator.

The documentation for this class was generated from the following file:

- [base.h](#)

6.147 `__gnu_parallel::_PseudoSequenceliterator<_Tp, _DifferenceTp>` Class Template Reference

```
#include <base.h>
```

Public Types

- typedef `_DifferenceTp` `_DifferenceType`

Public Member Functions

- `_PseudoSequenceliterator` (const `_Tp` &__val, `_DifferenceType` __pos)
- bool `operator!=` (const `_PseudoSequenceliterator` &__i2)
- const `_Tp` & `operator*` () const
- `_PseudoSequenceliterator` & `operator++` ()
- `_PseudoSequenceliterator` `operator++` (int)
- `_DifferenceType` `operator-` (const `_PseudoSequenceliterator` &__i2)
- bool `operator==` (const `_PseudoSequenceliterator` &__i2)
- const `_Tp` & `operator[]` (`_DifferenceType`) const

6.147.1 Detailed Description

```
template<typename _Tp, typename _DifferenceTp>
class __gnu_parallel::_PseudoSequenceIterator< _Tp, _DifferenceTp >
```

_Iterator associated with __gnu_parallel::_PseudoSequence. It features the usual random-access iterator functionality.

Parameters

<code>_Tp</code>	Sequence _M_value type.
<code>_DifferenceTp</code>	Sequence difference type.

The documentation for this class was generated from the following file:

- [base.h](#)

6.148 __gnu_parallel::_QSBThreadLocal< _RAIter > Struct Template Reference

```
#include <balanced_quicksort.h>
```

Public Types

- typedef `_TraitsType::difference_type` **DifferenceType**
- typedef `std::pair< _RAIter, _RAIter >` **Piece**
- typedef `std::iterator_traits< _RAIter >` **TraitsType**

Public Member Functions

- **QSBThreadLocal** (int __queue_size)

Public Attributes

- volatile `DifferenceType *` **M_elements_leftover**
- **Piece** **M_global**
- **Piece** **M_initial**
- `RestrictedBoundedConcurrentQueue< Piece >` **M_leftover_parts**
- `ThreadIndex` **M_num_threads**

6.148.1 Detailed Description

```
template<typename _RAIter>
struct __gnu_parallel::_QSBThreadLocal< _RAIter >
```

Information local to one thread in the parallel quicksort run.

6.148.2 Member Typedef Documentation

Piece

```
template<typename _RAIter >
typedef std::pair<_RAIter, _RAIter> __gnu_parallel::_QSBThreadLocal< _RAIter >::_Piece
```

Continuous part of the sequence, described by an iterator pair.

6.148.3 Constructor & Destructor Documentation

`_QSBThreadLocal()`

```
template<typename _RAIter >
__gnu_parallel::__QSBThreadLocal< _RAIter >::__QSBThreadLocal (
    int __queue_size ) [inline]
```

Constructor.

Parameters

<code>__queue_size</code>	size of the work-stealing queue.
---------------------------	----------------------------------

6.148.4 Member Data Documentation

`_M_elements_leftover`

```
template<typename _RAIter >
volatile _DifferenceType* __gnu_parallel::__QSBThreadLocal< _RAIter >::_M_elements_leftover
```

Pointer to a counter of elements left over to sort.

Referenced by [__gnu_parallel::__parallel_sort_qsb\(\)](#), [__gnu_parallel::__qsb_conquer\(\)](#), and [__gnu_parallel::__qsb_local_sort_with_helping\(\)](#).

`_M_global`

```
template<typename _RAIter >
_Piece __gnu_parallel::__QSBThreadLocal< _RAIter >::_M_global
```

The complete sequence to sort.

`_M_initial`

```
template<typename _RAIter >
_Piece __gnu_parallel::__QSBThreadLocal< _RAIter >::_M_initial
```

Initial piece to work on.

Referenced by [__gnu_parallel::__qsb_conquer\(\)](#), and [__gnu_parallel::__qsb_local_sort_with_helping\(\)](#).

`_M_leftover_parts`

```
template<typename _RAIter >
_RestrictedBoundedConcurrentQueue<_Piece> __gnu_parallel::__QSBThreadLocal< _RAIter >::_M_↔
leftover_parts
```

Work-stealing queue.

Referenced by [__gnu_parallel::__qsb_local_sort_with_helping\(\)](#).

`_M_num_threads`

```
template<typename _RAIter >
_ThreadIndex __gnu_parallel::__QSBThreadLocal< _RAIter >::_M_num_threads
```

Number of threads involved in this algorithm.

Referenced by [__gnu_parallel::__qsb_local_sort_with_helping\(\)](#).

The documentation for this struct was generated from the following file:

- [balanced_quicksort.h](#)

6.149 std::__detail::__Quoted_string<_String, _CharT > Struct Template Reference

```
#include <quoted_string.h>
```

Public Member Functions

- [_Quoted_string](#) (_String __str, _CharT __del, _CharT __esc)
- [_Quoted_string](#) & [operator=](#) ([_Quoted_string](#) &)=delete

Public Attributes

- [_CharT _M_delim](#)
- [_CharT _M_escape](#)
- [_String _M_string](#)

6.149.1 Detailed Description

```
template<typename _String, typename _CharT>
struct std::__detail::_Quoted_string< _String, _CharT >
```

Struct for delimited strings.

The documentation for this struct was generated from the following file:

- [quoted_string.h](#)

6.150 __gnu_parallel::_RandomNumber Class Reference

```
#include <random_number.h>
```

Public Member Functions

- [_RandomNumber](#) ()
- [_RandomNumber](#) (uint32_t __seed, uint64_t _M_supremum=0x100000000ULL)
- unsigned long [__genrand_bits](#) (int __bits)
- uint32_t [operator\(\)](#) ()
- uint32_t [operator\(\)](#) (uint64_t local_supremum)

6.150.1 Detailed Description

Random number generator, based on the Mersenne twister.

6.150.2 Constructor & Destructor Documentation

[_RandomNumber](#)() [1/2]

```
__gnu_parallel::_RandomNumber::_RandomNumber ( ) [inline]
```

Default constructor. Seed with 0.

[_RandomNumber](#)() [2/2]

```
__gnu_parallel::_RandomNumber::_RandomNumber (
    uint32_t __seed,
    uint64_t _M_supremum = 0x100000000ULL ) [inline]
```

Constructor.

Parameters

__seed	Random __seed.
_M_supremum	Generate integer random numbers in the interval [0,_M_supremum).

6.150.3 Member Function Documentation

`__genrand_bits()`

```
unsigned long __gnu_parallel::_RandomNumber::__genrand_bits (
    int __bits ) [inline]
```

Generate a number of random bits, run-time parameter.

Parameters

<code>__bits</code>	Number of bits to generate.
---------------------	-----------------------------

`operator>()` [1/2]

```
uint32_t __gnu_parallel::_RandomNumber::operator() ( ) [inline]
```

Generate unsigned random 32-bit integer.

`operator>()` [2/2]

```
uint32_t __gnu_parallel::_RandomNumber::operator() (
    uint64_t local_supremum ) [inline]
```

Generate unsigned random 32-bit integer in the interval [0,local_supremum).

The documentation for this class was generated from the following file:

- [random_number.h](#)

6.151 `__gnu_cxx::Relative_pointer_impl<_Tp>` Class Template Reference

```
#include <pointer.h>
```

Public Types

- typedef `_Tp element_type`

Public Member Functions

- `_Tp * get () const`
- bool `operator<` (const [_Relative_pointer_impl](#) &__rarg) const
- bool `operator==` (const [_Relative_pointer_impl](#) &__rarg) const
- void `set` (`_Tp *`__arg)

6.151.1 Detailed Description

```
template<typename _Tp>
```

```
class __gnu_cxx::Relative_pointer_impl<_Tp>
```

A storage policy for use with `_Pointer_adapter<>` which stores the pointer's address as an offset value which is relative to its own address.

This is intended for pointers within shared memory regions which might be mapped at different addresses by different processes. For null pointers, a value of 1 is used. (0 is legitimate sometimes for nodes in circularly linked lists) This value was chosen as the least likely to generate an incorrect null, As there is no reason why any normal pointer would point 1 byte into its own pointer address.

The documentation for this class was generated from the following file:

- [pointer.h](#)

6.152 `__gnu_cxx::Relative_pointer_impl< const _Tp >` Class Template Reference

```
#include <pointer.h>
```

Public Types

- typedef const _Tp **element_type**

Public Member Functions

- const _Tp * **get** () const
- bool **operator<** (const [_Relative_pointer_impl](#) &__rarg) const
- bool **operator==** (const [_Relative_pointer_impl](#) &__rarg) const
- void **set** (const _Tp *__arg)

6.152.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::Relative_pointer_impl< const _Tp >
```

`Relative_pointer_impl` needs a specialization for const T because of the casting done during pointer arithmetic. The documentation for this class was generated from the following file:

- [pointer.h](#)

6.153 `__gnu_parallel::RestrictedBoundedConcurrentQueue< _Tp >` Class Template Reference

```
#include <queue.h>
```

Public Member Functions

- [_RestrictedBoundedConcurrentQueue](#) ([_SequenceIndex](#) __max_size)
- [~_RestrictedBoundedConcurrentQueue](#) ()
- bool [pop_back](#) (_Tp &__t)
- bool [pop_front](#) (_Tp &__t)
- void [push_front](#) (const _Tp &__t)

6.153.1 Detailed Description

```
template<typename _Tp>
class __gnu_parallel::RestrictedBoundedConcurrentQueue< _Tp >
```

Double-ended queue of bounded size, allowing lock-free atomic access. `push_front()` and `pop_front()` must not be called concurrently to each other, while `pop_back()` can be called concurrently at all times. `empty()`, `size()`, and `top()` are intentionally not provided. Calling them would not make sense in a concurrent setting.

Parameters

<code>_Tp</code>	Contained element type.
------------------	-------------------------

6.153.2 Constructor & Destructor Documentation

`_RestrictedBoundedConcurrentQueue()`

```
template<typename _Tp >
__gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::_RestrictedBoundedConcurrentQueue (
    __SequenceIndex __max_size ) [inline]
```

Constructor. Not to be called concurrent, of course.

Parameters

<code>__max_size</code>	Maximal number of elements to be contained.
-------------------------	---

References [__gnu_parallel::__encode2\(\)](#).

`~_RestrictedBoundedConcurrentQueue()`

```
template<typename _Tp >
__gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::~~_RestrictedBoundedConcurrentQueue ( )
[inline]
```

Destructor. Not to be called concurrent, of course.

6.153.3 Member Function Documentation

`pop_back()`

```
template<typename _Tp >
bool __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::pop_back (
    _Tp & __t ) [inline]
```

Pops one element from the queue at the front end. Must not be called concurrently with `pop_front()`.

References [__gnu_parallel::__compare_and_swap\(\)](#), [__gnu_parallel::__decode2\(\)](#), and [__gnu_parallel::__encode2\(\)](#).

`pop_front()`

```
template<typename _Tp >
bool __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::pop_front (
    _Tp & __t ) [inline]
```

Pops one element from the queue at the front end. Must not be called concurrently with `pop_front()`.

References [__gnu_parallel::__compare_and_swap\(\)](#), [__gnu_parallel::__decode2\(\)](#), and [__gnu_parallel::__encode2\(\)](#).

`push_front()`

```
template<typename _Tp >
void __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::push_front (
    const _Tp & __t ) [inline]
```

Pushes one element into the queue at the front end. Must not be called concurrently with `pop_front()`.

References [__gnu_parallel::__decode2\(\)](#), [__gnu_parallel::__encode2\(\)](#), and [__gnu_parallel::__fetch_and_add\(\)](#).

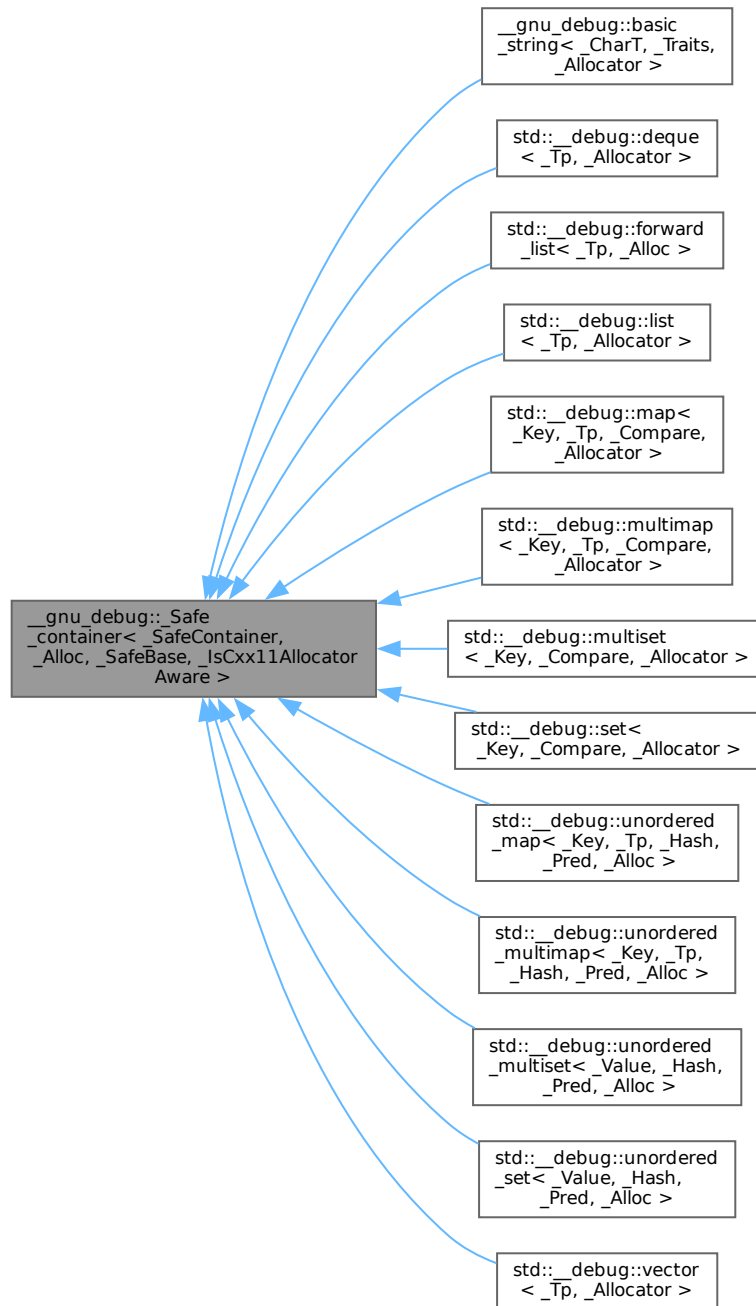
The documentation for this class was generated from the following file:

- [queue.h](#)

6.154 `__gnu_debug::_Safe_container<_SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >` Class Template Reference

```
#include <safe_container.h>
```

Inheritance diagram for `__gnu_debug::_Safe_container<_SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware>`:



Protected Member Functions

- `_Safe_container` (`_Safe_container` &&)=default
- `_Safe_container` (`_Safe_container` &&__x, const `_Alloc` &__a)

- `_Safe_container` (const `_Safe_container` &)=default
- void `_M_swap` (`_Safe_container` &__x) noexcept
- `_Safe_container` & `operator=` (`_Safe_container` && __x) noexcept
- `_Safe_container` & `operator=` (const `_Safe_container` &) noexcept

6.154.1 Detailed Description

template<typename `_SafeContainer`, typename `_Alloc`, template< typename > class `_SafeBase`, bool `_IsCxx11AllocatorAware` = true>

class `__gnu_debug::_Safe_container`< `_SafeContainer`, `_Alloc`, `_SafeBase`, `_IsCxx11AllocatorAware` >

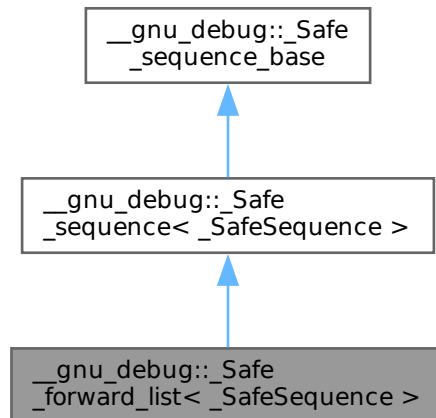
Safe class dealing with some allocator dependent operations.

The documentation for this class was generated from the following file:

- [safe_container.h](#)

6.155 __gnu_debug::_Safe_forward_list<_SafeSequence> Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_forward_list<_SafeSequence>`:



Public Member Functions

- void `_M_invalidate_if` (`_Predicate` __pred)
- void `_M_transfer_from_if` (`_Safe_sequence` &__from, `_Predicate` __pred)

Public Attributes

- `_Safe_iterator_base` * `_M_const_iterators`
- `_Safe_iterator_base` * `_M_iterators`
- unsigned int `_M_version`

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_invalidate_all](#) ()
- void [_M_invalidate_all](#) () const
- void [_M_revalidate_singular](#) ()
- void [_M_swap](#) ([_Safe_sequence_base](#) &) noexcept

6.155.1 Detailed Description

```
template<typename _SafeSequence>
class __gnu_debug::_Safe_forward_list< _SafeSequence >
```

Special iterators swap and invalidation for forward_list because of the before_begin iterator.

6.155.2 Member Function Documentation

[_M_detach_all\(\)](#)

```
void __gnu_debug::_Safe_sequence_base::_M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

Referenced by [__gnu_debug::_Safe_sequence_base::~~_Safe_sequence_base\(\)](#).

[_M_detach_singular\(\)](#)

```
void __gnu_debug::_Safe_sequence_base::_M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

[_M_get_mutex\(\)](#)

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_sequence_base::_M_get_mutex ( ) throw ( ) [protected],
[inherited]
```

For use in _Safe_sequence.

Referenced by [__gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if\(\)](#).

[_M_invalidate_all\(\)](#)

```
void __gnu_debug::_Safe_sequence_base::_M_invalidate_all ( ) const [inline], [protected], [inherited]
```

Invalidates all iterators.

References [__gnu_debug::_Safe_sequence_base::_M_version](#).

[_M_invalidate_if\(\)](#)

```
void __gnu_debug::_Safe_sequence< _SafeSequence >::_M_invalidate_if (
    _Predicate __pred ) [inherited]
```

Invalidates all iterators x that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

`_M_revalidate_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ( ) [protected], [inherited]
```

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_transfer_from_if()`

```
void __gnu_debug::_Safe_sequence<_SafeSequence>::_M_transfer_from_if (
    _Safe_sequence<_SafeSequence> & __from,
    _Predicate __pred ) [inherited]
```

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

6.155.3 Member Data Documentation**`_M_const_iterators`**

```
_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_const_iterators [inherited]
```

The list of constant iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_iterators`

```
_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators [inherited]
```

The list of mutable iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_version`

```
unsigned int __gnu_debug::_Safe_sequence_base::_M_version [mutable], [inherited]
```

The container version number. This number may never be 0.

Referenced by `__gnu_debug::_Safe_sequence_base::_M_invalidate_all()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>`.

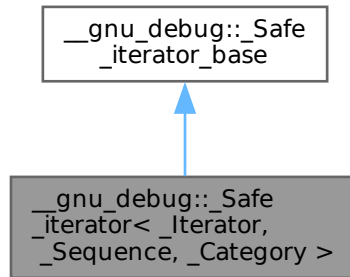
The documentation for this class was generated from the following file:

- [debug/forward_list](#)

6.156 `__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>` Class Template Reference

```
#include <safe_iterator.h>
```

Inheritance diagram for `__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category >`:



Public Types

- typedef `_Safe_iterator` `<_Iterator, _Sequence, iterator_category >_Self`
- typedef `_Traits::difference_type` **difference_type**
- typedef `_Traits::iterator_category` **iterator_category**
- using **iterator_concept** = `std::__detail::__iter_concept<_Iterator >`
- typedef `_Iterator` **iterator_type**
- typedef `_Traits::pointer` **pointer**
- typedef `_Traits::reference` **reference**
- typedef `_Traits::value_type` **value_type**

Public Member Functions

- `_Safe_iterator` () noexcept
- `_Safe_iterator` (`_Iterator __i`, `const _Safe_sequence_base *__seq`) noexcept
- `_Safe_iterator` (`_Safe_iterator &&__x`) noexcept
- `_Safe_iterator` (`const _Safe_iterator &__x`) noexcept
- `template<typename _MutableIterator >`
`_Safe_iterator` (`const _Safe_iterator<_MutableIterator, _Sequence, typename __gnu_cxx::__enable_if<_IsConstant::__value &&std::__are_same<_MutableIterator, _OtherIterator >::__value, _Category >::__type >`
`&__x`) noexcept
- `void _M_attach` (`_Safe_sequence_base *__seq`)
- `void _M_attach_single` (`_Safe_sequence_base *__seq`)
- `bool _M_attached_to` (`const _Safe_sequence_base *__seq`) const
- `bool _M_before_dereferenceable` () const
- `template<typename _Diff >`
`bool _M_can_advance` (`const std::pair<_Diff, _Distance_precision > &__dist`, `int __way`) const
- `bool _M_can_advance` (`difference_type __n`, `bool __strict=false`) const
- `bool _M_can_compare` (`const _Safe_iterator_base &__x`) const throw ()
- `bool _M_dereferenceable` () const
- `void _M_detach_single` () throw ()
- `_Distance_traits<_Iterator >::__type _M_get_distance_from_begin` () const
- `_Distance_traits<_Iterator >::__type _M_get_distance_to` (`const _Safe_iterator &__rhs`) const
- `_Distance_traits<_Iterator >::__type _M_get_distance_to_end` () const

- `__gnu_cxx::__conditional_type<_IsConstant::__value, const_Sequence *, _Sequence * >::__type _M_get_↵
sequence () const`
- `bool _M_incrementable () const`
- `void _M_invalidate ()`
- `bool _M_is_before_begin () const`
- `bool _M_is_begin () const`
- `bool _M_is_beginnest () const`
- `bool _M_is_end () const`
- `void _M_reset () throw ()`
- `bool _M_singular () const throw ()`
- `void _M_unlink () throw ()`
- `bool _M_valid_range (const _Safe_iterator &__rhs, std::pair< difference_type, _Distance_precision > &__dist,
bool __check_dereferenceable=true) const`
- `const _Iterator & base () const noexcept`
- `_Iterator & base () noexcept`
- `operator _Iterator () const noexcept`
- `reference operator* () const noexcept`
- `_Safe_iterator & operator++ () noexcept`
- `_Safe_iterator operator++ (int) noexcept`
- `pointer operator-> () const noexcept`
- `_Safe_iterator & operator= (_Safe_iterator &&__x) noexcept`
- `_Safe_iterator & operator= (const _Safe_iterator &__x) noexcept`

Static Public Member Functions

- `static constexpr bool _S_constant ()`

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- `unsigned int _M_version`

Protected Types

- `typedef std::__are_same< typename _Sequence::__Base::const_iterator, _Iterator > _IsConstant`
- `typedef __gnu_cxx::__conditional_type< _IsConstant::__value, typename _Sequence::__Base::iterator, typename_↵
_Sequence::__Base::const_iterator >::__type _OtherIterator`

Protected Member Functions

- `_Safe_iterator (_Iterator __i, _Safe_sequence_base * __seq, _Attach_single) noexcept`
- `void _M_attach (_Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw ()`
- `void _M_detach ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`

Friends

- `template<typename _IterR >`
`bool operator!= (const _Self &__lhs, const _Safe_iterator< _IterR, _Sequence, iterator_category > &__rhs) noexcept`
- `bool operator!= (const _Self &__lhs, const _Self &__rhs) noexcept`
- `template<typename _IterR >`
`bool operator== (const _Self &__lhs, const _Safe_iterator< _IterR, _Sequence, iterator_category > &__rhs) noexcept`
- `bool operator== (const _Self &__lhs, const _Self &__rhs) noexcept`

6.156.1 Detailed Description

```
template<typename _Iterator, typename _Sequence, typename _Category = typename std::iterator_traits<_↵
Iterator>::iterator_category>
class \_\_gnu\_debug::\_Safe\_iterator< \_Iterator, \_Sequence, \_Category >
```

Safe iterator wrapper.

The class template `_Safe_iterator` is a wrapper around an iterator that tracks the iterator's movement among sequences and checks that operations performed on the "safe" iterator are legal. In addition to the basic iterator operations (which are validated, and then passed to the underlying iterator), `_Safe_iterator` has member functions for iterator invalidation, attaching/detaching the iterator from sequences, and querying the iterator's state.

Note that `_Iterator` must be the first base class so that it gets initialized before the iterator is being attached to the container's list of iterators and it is being detached before `_Iterator` get destroyed. Otherwise it would result in a data race.

6.156.2 Constructor & Destructor Documentation

[_Safe_iterator\(\)](#) [1/5]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_↵
traits<_Iterator>::iterator_category>
\_\_gnu\_debug::\_Safe\_iterator< \_Iterator, \_Sequence, \_Category >::_Safe_iterator ( ) [inline],
[noexcept]
```

Postcondition

the iterator is singular and unattached

Referenced by [__gnu_debug::_Safe_iterator](#)< [_Iterator](#), [_Sequence](#), [_Category](#) >::operator++().

[_Safe_iterator\(\)](#) [2/5]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_↵
traits<_Iterator>::iterator_category>
\_\_gnu\_debug::\_Safe\_iterator< \_Iterator, \_Sequence, \_Category >::_Safe_iterator (
    \_Iterator __i,
    const \_Safe\_sequence\_base * __seq ) [inline], [noexcept]
```

Safe iterator construction from an unsafe iterator and its sequence.

Precondition

`seq` is not NULL

Postcondition

this is not singular

References [__gnu_debug::_Safe_iterator_base::M_singular\(\)](#).

`_Safe_iterator()` [3/5]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator (
    const _Safe_iterator< _Iterator, _Sequence, _Category > & __x ) [inline], [noexcept]
```

Copy construction.

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach\(\)](#).

`_Safe_iterator()` [4/5]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator (
    _Safe_iterator< _Iterator, _Sequence, _Category > && __x ) [inline], [noexcept]
```

Move construction.

Postcondition

`__x` is singular and unattached

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_detach\(\)](#), and [std::swap\(\)](#).

`_Safe_iterator()` [5/5]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
template<typename _MutableIterator >
__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator (
    const _Safe_iterator< _MutableIterator, _Sequence, typename __gnu_cxx::__enable_if<
        _IsConstant::__value &&std::__are_same< _MutableIterator, _OtherIterator >::__value, _Category
    >::__type > & __x ) [inline], [noexcept]
```

Converting constructor from a mutable iterator to a constant iterator.

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach\(\)](#).

6.156.3 Member Function Documentation**`_M_attach()`** [1/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
void __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_attach (
    _Safe_sequence_base * __seq ) [inline]
```

Attach iterator to the given sequence.

References [__gnu_debug::_Safe_iterator_base::_M_attach\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_detach\(\)](#).
Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(const_iterator\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(const_iterator, const_iterator\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#).

`_M_attach()` [2/2]

```
void __gnu_debug::_Safe_iterator_base::_M_attach (
    _Safe_sequence_base * __seq,
    bool __constant ) [protected], [inherited]
```

Attaches this iterator to the given sequence, detaching it from whatever sequence it was attached to originally. If the new sequence is the NULL pointer, the iterator is left unattached.

Referenced by [__gnu_debug::_Safe_iterator_base::_Safe_iterator_base\(\)](#), [__gnu_debug::_Safe_iterator_base::_Safe_iterator_base\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach\(\)](#).

M_attach_single() [1/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
void __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_attach_single (
    _Safe_sequence_base * __seq ) [inline]
```

Likewise, but not thread-safe.

References [__gnu_debug::_Safe_iterator_base::_M_attach_single\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach\(\)](#).

M_attach_single() [2/2]

```
void __gnu_debug::_Safe_iterator_base::_M_attach_single (
    _Safe_sequence_base * __seq,
    bool __constant ) throw ( ) [protected], [inherited]
```

Likewise, but not thread-safe.

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach_single\(\)](#).

M_attached_to()

```
bool __gnu_debug::_Safe_iterator_base::_M_attached_to (
    const _Safe_sequence_base * __seq ) const [inline], [inherited]
```

Determines if we are attached to the given sequence.

References [__gnu_debug::_Safe_iterator_base::_M_sequence](#).

M_before_dereferenceable()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
bool __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_before_dereferenceable (
    ) const [inline]
```

Is the iterator before a dereferenceable one?

References [__gnu_debug::_base\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_incrementable\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_base\(\)](#).

M_can_compare()

```
bool __gnu_debug::_Safe_iterator_base::_M_can_compare (
    const _Safe_iterator_base & __x ) const throw ( ) [inherited]
```

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

M_dereferenceable()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
bool __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_dereferenceable ( ) const [inline]
```

Is the iterator dereferenceable?

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_is_before_begin\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_is_before_end\(\)](#), and [__gnu_debug::_Safe_iterator_base::_M_singular\(\)](#).

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator*\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#).

`_M_detach()`

```
void __gnu_debug::__Safe_iterator_base::_M_detach ( ) [protected], [inherited]
```

Detach the iterator for whatever sequence it is attached to, if any.

Referenced by [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#).

`_M_detach_single()`

```
void __gnu_debug::__Safe_iterator_base::_M_detach_single ( ) throw ( ) [inherited]
```

Likewise, but not thread-safe.

Referenced by [__gnu_debug::__Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#).

`_M_get_mutex()`

```
__gnu_cxx::__mutex & __gnu_debug::__Safe_iterator_base::_M_get_mutex ( ) throw ( ) [protected], [inherited]
```

For use in `_Safe_iterator`.

Referenced by [__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#).

`_M_incrementable()`

```
template<typename _Iterator, typename _Sequence, typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
bool __gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_incrementable ( ) const [inline]
```

Is the iterator incrementable?

References [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_is_end\(\)](#), and [__gnu_debug::__Safe_iterator_base::_M_is_end\(\)](#).

Referenced by [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_before_dereferenceable\(\)](#), [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), and [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#).

`_M_invalidate()`

```
void __gnu_debug::__Safe_iterator_base::_M_invalidate ( ) [inline], [inherited]
```

Invalidate the iterator, making it singular.

References [__gnu_debug::__Safe_iterator_base::_M_version](#).

`_M_is_before_begin()`

```
template<typename _Iterator, typename _Sequence, typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
bool __gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_is_before_begin ( ) const [inline]
```

Is this iterator equal to the sequence's `before_begin()` iterator if any?

Referenced by [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_dereferenceable\(\)](#).

`_M_is_begin()`

```
template<typename _Iterator, typename _Sequence, typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
```

```
bool __gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::_M_is_begin ( ) const [inline]
```

Is this iterator equal to the sequence's `begin()` iterator?

References [__gnu_debug::__Safe_iterator<_Iterator, _Sequence, _Category>::base\(\)](#).

M_is_beginnest()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
bool __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_is_beginnest ( ) const
[inline]
```

Is this iterator equal to the sequence's before_begin() iterator if any or begin() otherwise?

M_is_end()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
bool __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_is_end ( ) const [inline]
```

Is this iterator equal to the sequence's end() iterator?

References [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::base\(\)](#).

Referenced by [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_incrementable\(\)](#).

M_reset()

```
void __gnu_debug::_Safe_iterator_base::_M_reset ( ) throw ( ) [inherited]
```

Reset all member variables

M_singular()

```
bool __gnu_debug::_Safe_iterator_base::_M_singular ( ) const throw ( ) [inherited]
```

Is this iterator singular?

Referenced by [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence, _Category >::_Safe_local_iterator\(\)](#), [__gnu_debug::_check_singular_aux\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence >::_M_incrementable\(\)](#), and [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_incrementable\(\)](#).

M_unlink()

```
void __gnu_debug::_Safe_iterator_base::_M_unlink ( ) throw ( ) [inline], [inherited]
```

Unlink itself

References [__gnu_debug::_Safe_iterator_base::_M_next](#), and [__gnu_debug::_Safe_iterator_base::_M_prior](#).

S_constant()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
static constexpr bool __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_S_constant
( ) [inline], [static], [constexpr]
```

Determine if this is a constant iterator.

Referenced by [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_attach\(\)](#), and [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_is_begin\(\)](#).

base()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
_Iterator & __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::base ( ) [inline],
[noexcept]
```

Return the underlying iterator.

Referenced by [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_incrementable\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_is_begin\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_unlink\(\)](#), and [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::_M_unlink\(\)](#).

[__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator*\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::base\(\)](#).

operator _Iterator()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator _Iterator ( ) const
[inline], [noexcept]
```

Conversion to underlying non-debug iterator to allow better interaction with non-debug containers.

operator*()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
reference __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator* ( ) const
[inline], [noexcept]
```

Iterator dereference.

Precondition

iterator is dereferenceable

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::base\(\)](#).

operator++() [1/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
_Safe_iterator & __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator++ ( )
[inline], [noexcept]
```

Iterator preincrement.

Precondition

iterator is incrementable

References [__gnu_debug::_Safe_iterator_base::M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::M_increment\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::base\(\)](#).

operator++() [2/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
_Safe_iterator __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator++ (
    int ) [inline], [noexcept]
```

Iterator postincrement.

Precondition

iterator is incrementable

References [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_iterator_base::M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::M_incrementable\(\)](#), [__gnu_debug::_Safe_iterator_base::M_incrementable\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::base\(\)](#).

operator->()

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
pointer __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator-> ( ) const
[inline], [noexcept]
```

Iterator dereference.

Precondition

iterator is dereferenceable

References [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_iterator](#)

operator=() [1/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
_Safe_iterator & __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator= (
    _Safe_iterator< _Iterator, _Sequence, _Category > && __x ) [inline], [noexcept]
```

Move assignment.

Postcondition

__x is singular and unattached

References [std::_addressof\(\)](#), [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::M_attach\(\)](#), [__gnu_debug::_Safe_iterator_base::M_detach\(\)](#), [__gnu_debug::_Safe_iterator_base::M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator_base::M_version](#), and [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::base\(\)](#).

operator=() [2/2]

```
template<typename _Iterator , typename _Sequence , typename _Category = typename std::iterator_traits<_Iterator>::iterator_category>
_Safe_iterator & __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::operator= (
    const _Safe_iterator< _Iterator, _Sequence, _Category > & __x ) [inline], [noexcept]
```

Copy assignment.

References [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::M_attach\(\)](#), [__gnu_debug::_Safe_iterator_base::M_detach\(\)](#), [__gnu_debug::_Safe_iterator_base::M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator_base::M_sequence](#), [__gnu_debug::_Safe_iterator_base::M_version](#), and [__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >::base\(\)](#).

6.156.4 Member Data Documentation**M_next**

[_Safe_iterator_base](#)* [__gnu_debug::_Safe_iterator_base::M_next](#) [inherited]

Pointer to the next iterator in the sequence's list of iterators. Only valid when [_M_sequence](#) != NULL.

Referenced by [__gnu_debug::_Safe_sequence< _SafeSequence >::M_invalidate_if\(\)](#), [__gnu_debug::_Safe_sequence< _Sequence >::M_invalidate_if\(\)](#), and [__gnu_debug::_Safe_iterator_base::M_unlink\(\)](#).

M_prior

[_Safe_iterator_base](#)* [__gnu_debug::_Safe_iterator_base::M_prior](#) [inherited]

Pointer to the previous iterator in the sequence's list of iterators. Only valid when [_M_sequence](#) != NULL.

Referenced by [__gnu_debug::_Safe_sequence< _SafeSequence >::M_invalidate_if\(\)](#), [__gnu_debug::_Safe_sequence< _Sequence >::M_invalidate_if\(\)](#), and [__gnu_debug::_Safe_iterator_base::M_unlink\(\)](#).

`_M_sequence`

`_Safe_sequence_base* __gnu_debug::Safe_iterator_base::_M_sequence` [inherited]

The sequence this iterator references; may be NULL to indicate a singular iterator.

Referenced by `__gnu_debug::Safe_iterator_base::Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator_base::Safe_local_iterator_base()`, `__gnu_debug::Safe_iterator_base::_M_attached_to()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator++()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator++()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`.

`_M_version`

`unsigned int __gnu_debug::Safe_iterator_base::_M_version` [inherited]

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Referenced by `__gnu_debug::Safe_iterator_base::_M_invalidate()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`.

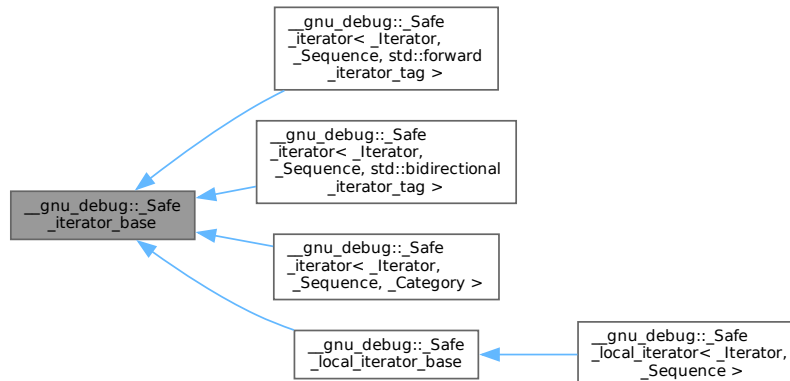
The documentation for this class was generated from the following files:

- `boost_concept_check.h`
- `safe_iterator.h`
- `safe_iterator.tcc`

6.157 `__gnu_debug::Safe_iterator_base` Class Reference

```
#include <safe_base.h>
```

Inheritance diagram for `__gnu_debug::Safe_iterator_base`:



Public Member Functions

- `bool _M_attached_to (const _Safe_sequence_base * __seq) const`
- `bool _M_can_compare (const _Safe_iterator_base & __x) const throw ()`
- `void _M_detach_single () throw ()`
- `void _M_invalidate ()`
- `void _M_reset () throw ()`

- `bool _M_singular () const throw ()`
- `void _M_unlink () throw ()`

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- `unsigned int _M_version`

Protected Member Functions

- `_Safe_iterator_base ()`
- `_Safe_iterator_base (const _Safe_iterator_base &__x, bool __constant)`
- `_Safe_iterator_base (const _Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach (_Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw ()`
- `void _M_detach ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`

Friends

- `class _Safe_sequence_base`

6.157.1 Detailed Description

Basic functionality for a *safe* iterator.

The `_Safe_iterator_base` base class implements the functionality of a safe iterator that is not specific to a particular iterator type. It contains a pointer back to the sequence it references along with iterator version information and pointers to form a doubly-linked list of iterators referenced by the container.

This class must not perform any operations that can throw an exception, or the exception guarantees of derived iterators will be broken.

6.157.2 Constructor & Destructor Documentation

`_Safe_iterator_base()` [1/3]

```
__gnu_debug::_Safe_iterator_base::_Safe_iterator_base ( ) [inline], [protected]
```

Initializes the iterator and makes it singular.

`_Safe_iterator_base()` [2/3]

```
__gnu_debug::_Safe_iterator_base::_Safe_iterator_base (
    const _Safe_sequence_base * __seq,
    bool __constant ) [inline], [protected]
```

Initialize the iterator to reference the sequence pointed to by `__seq`. `__constant` is true when we are initializing a constant iterator, and false if it is a mutable iterator. Note that `__seq` may be NULL, in which case the iterator will be singular. Otherwise, the iterator will reference `__seq` and be nonsingular.

References `_M_attach()`.

`_Safe_iterator_base()` [3/3]

```
__gnu_debug::_Safe_iterator_base::_Safe_iterator_base (
    const \_Safe\_iterator\_base & __x,
    bool __constant ) [inline], [protected]
```

Initializes the iterator to reference the same sequence that `__x` does. `__constant` is true if this is a constant iterator, and false if it is mutable.

References [_M_attach\(\)](#), and [_M_sequence](#).

6.157.3 Member Function Documentation**`_M_attach()`**

```
void __gnu_debug::_Safe_iterator_base::_M_attach (
    \_Safe\_sequence\_base * __seq,
    bool __constant ) [protected]
```

Attaches this iterator to the given sequence, detaching it from whatever sequence it was attached to originally. If the new sequence is the NULL pointer, the iterator is left unattached.

Referenced by [_Safe_iterator_base\(\)](#), [_Safe_iterator_base\(\)](#), and [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M](#)

`_M_attach_single()`

```
void __gnu_debug::_Safe_iterator_base::_M_attach_single (
    \_Safe\_sequence\_base * __seq,
    bool __constant ) throw ( ) [protected]
```

Likewise, but not thread-safe.

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_attach_single\(\)](#).

`_M_attached_to()`

```
bool __gnu_debug::_Safe_iterator_base::_M_attached_to (
    const \_Safe\_sequence\_base * __seq ) const [inline]
```

Determines if we are attached to the given sequence.

References [_M_sequence](#).

`_M_can_compare()`

```
bool __gnu_debug::_Safe_iterator_base::_M_can_compare (
    const \_Safe\_iterator\_base & __x ) const throw ( )
```

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

`_M_detach()`

```
void __gnu_debug::_Safe_iterator_base::_M_detach ( ) [protected]
```

Detach the iterator for whatever sequence it is attached to, if any.

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::_Safe_iterator<_Itera](#)

`_M_detach_single()`

```
void __gnu_debug::_Safe_iterator_base::_M_detach_single ( ) throw ( )
```

Likewise, but not thread-safe.

Referenced by [__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#).

M_get_mutex()

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_iterator_base::_M_get_mutex ( ) throw ( ) [protected]
```

For use in `_Safe_iterator`.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#).

M_invalidate()

```
void __gnu_debug::_Safe_iterator_base::_M_invalidate ( ) [inline]
```

Invalidate the iterator, making it singular.

References [_M_version](#).

M_reset()

```
void __gnu_debug::_Safe_iterator_base::_M_reset ( ) throw ( )
```

Reset all member variables

M_singular()

```
bool __gnu_debug::_Safe_iterator_base::_M_singular ( ) const throw ( )
```

Is this iterator singular?

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::check_singular_aux\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_incrementable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable\(\)](#).

M_unlink()

```
void __gnu_debug::_Safe_iterator_base::_M_unlink ( ) throw ( ) [inline]
```

Unlink itself

References [_M_next](#), and [_M_prior](#).

6.157.4 Member Data Documentation**M_next**

```
\_Safe\_iterator\_base\* __gnu_debug::_Safe_iterator_base::_M_next
```

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Referenced by [__gnu_debug::_Safe_sequence<_SafeSequence>::_M_invalidate_if\(\)](#), [__gnu_debug::_Safe_sequence<_Sequence>::_M_invalidate_if\(\)](#), and [_M_unlink\(\)](#).

M_prior

```
\_Safe\_iterator\_base\* __gnu_debug::_Safe_iterator_base::_M_prior
```

Pointer to the previous iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Referenced by [__gnu_debug::_Safe_sequence<_SafeSequence>::_M_invalidate_if\(\)](#), [__gnu_debug::_Safe_sequence<_Sequence>::_M_invalidate_if\(\)](#), and [_M_unlink\(\)](#).

M_sequence

```
\_Safe\_sequence\_base\* __gnu_debug::_Safe_iterator_base::_M_sequence
```

The sequence this iterator references; may be NULL to indicate a singular iterator.

Referenced by `_Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator_base::Safe_local_iterator_base()`, `_M_attached_to()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator++()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator++()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

`_M_version`

```
unsigned int __gnu_debug::Safe_iterator_base::_M_version
```

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Referenced by `_M_invalidate()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator++()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

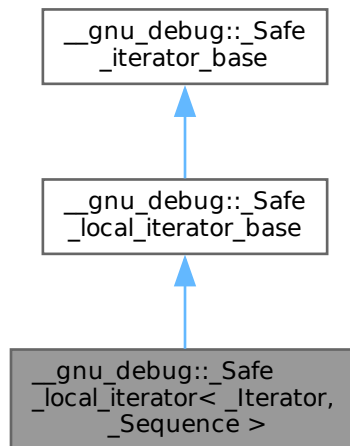
The documentation for this class was generated from the following file:

- [safe_base.h](#)

6.158 `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>` Class Template Reference

```
#include <safe_local_iterator.h>
```

Inheritance diagram for `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>`:



Public Types

- `typedef _Traits::difference_type` **difference_type**
- `typedef _Traits::iterator_category` **iterator_category**
- `typedef _Iterator` **iterator_type**
- `typedef _Traits::pointer` **pointer**
- `typedef _Traits::reference` **reference**
- `typedef _Traits::value_type` **value_type**

Public Member Functions

- [_Safe_local_iterator](#) () noexcept
- [_Safe_local_iterator](#) (_Iterator __i, const [_Safe_sequence_base](#) * __cont)
- [_Safe_local_iterator](#) ([_Safe_local_iterator](#) && __x) noexcept
- [_Safe_local_iterator](#) (const [_Safe_local_iterator](#) & __x) noexcept
- template<typename [_MutableIterator](#) >
[_Safe_local_iterator](#) (const [_Safe_local_iterator](#)< [_MutableIterator](#), typename [__gnu_cxx::__enable_if](#)< [_IsConstant](#)::__value &&std::__are_same< [_MutableIterator](#), [_OtherIterator](#) >::__value, [_Sequence](#) >::__type > & __x) noexcept
- void [_M_attach](#) ([_Safe_sequence_base](#) * __seq)
- void [_M_attach_single](#) ([_Safe_sequence_base](#) * __seq)
- bool [_M_attached_to](#) (const [_Safe_sequence_base](#) * __seq) const
- bool [_M_can_compare](#) (const [_Safe_iterator_base](#) & __x) const throw ()
- bool [_M_dereferenceable](#) () const
- [_Distance_traits](#)< [_Iterator](#) >::__type [_M_get_distance_to](#) (const [_Safe_local_iterator](#) & __rhs) const
- [__gnu_cxx::__conditional_type](#)< [_IsConstant](#)::__value, const [_Sequence](#) *, [_Sequence](#) * >::__type [_M_get_sequence](#) () const
- template<typename [_Other](#) >
bool [_M_in_same_bucket](#) (const [_Safe_local_iterator](#)< [_Other](#), [_Sequence](#) > & __other) const
- bool [_M_incrementable](#) () const
- void [_M_invalidate](#) ()
- bool [_M_is_begin](#) () const
- bool [_M_is_end](#) () const
- void [_M_reset](#) () throw ()
- bool [_M_singular](#) () const throw ()
- void [_M_unlink](#) () throw ()
- bool [_M_valid_range](#) (const [_Safe_local_iterator](#) & __rhs, [std::pair](#)< difference_type, [_Distance_precision](#) > & __dist_info) const
- const [_Iterator](#) & **base** () const noexcept
- [_Iterator](#) & **base** () noexcept
- size_type [bucket](#) () const
- [operator _Iterator](#) () const
- reference [operator*](#) () const
- [_Safe_local_iterator](#) & [operator++](#) ()
- [_Safe_local_iterator](#) [operator++](#) (int)
- pointer [operator->](#) () const
- [_Safe_local_iterator](#) & [operator=](#) ([_Safe_local_iterator](#) && __x) noexcept
- [_Safe_local_iterator](#) & [operator=](#) (const [_Safe_local_iterator](#) & __x)

Static Public Member Functions

- static constexpr bool [_S_constant](#) ()

Public Attributes

- [_Safe_iterator_base](#) * [_M_next](#)
- [_Safe_iterator_base](#) * [_M_prior](#)
- [_Safe_sequence_base](#) * [_M_sequence](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void `_M_attach` (`_Safe_sequence_base * __seq`, `bool __constant`)
- void `_M_attach_single` (`_Safe_sequence_base * __seq`, `bool __constant`) throw ()
- void `_M_detach` ()
- void `_M_detach_single` () throw ()
- `_Safe_unordered_container_base * _M_get_container` () const noexcept
- `__gnu_cxx::__mutex & _M_get_mutex` () throw ()

Friends

- `bool operator!=` (const `_Self` & __lhs, const `_OtherSelf` & __rhs) noexcept
- `bool operator!=` (const `_Self` & __lhs, const `_Self` & __rhs) noexcept
- `bool operator==` (const `_Self` & __lhs, const `_OtherSelf` & __rhs) noexcept
- `bool operator==` (const `_Self` & __lhs, const `_Self` & __rhs) noexcept

6.158.1 Detailed Description

`template<typename _Iterator, typename _Sequence>`
`class __gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>`

Safe iterator wrapper.

The class template `_Safe_local_iterator` is a wrapper around an iterator that tracks the iterator's movement among sequences and checks that operations performed on the "safe" iterator are legal. In addition to the basic iterator operations (which are validated, and then passed to the underlying iterator), `_Safe_local_iterator` has member functions for iterator invalidation, attaching/detaching the iterator from sequences, and querying the iterator's state.

6.158.2 Constructor & Destructor Documentation**`_Safe_local_iterator()` [1/5]**

```
template<typename _Iterator , typename _Sequence >
__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::__Safe_local_iterator ( ) [inline],
[noexcept]
```

Postcondition

the iterator is singular and unattached

Referenced by `__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::operator++()`.

`_Safe_local_iterator()` [2/5]

```
template<typename _Iterator , typename _Sequence >
__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::__Safe_local_iterator (
    _Iterator __i,
    const _Safe_sequence_base * __cont ) [inline]
```

Safe iterator construction from an unsafe iterator and its sequence.

Precondition

`seq` is not NULL

Postcondition

this is not singular

References `__gnu_debug::__Safe_iterator_base::_M_singular()`.

_Safe_local_iterator() [3/5]

```
template<typename _Iterator , typename _Sequence >
__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator (
    const _Safe_local_iterator< _Iterator, _Sequence > & __x ) [inline], [noexcept]
```

Copy construction.

References [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach\(\)](#).

_Safe_local_iterator() [4/5]

```
template<typename _Iterator , typename _Sequence >
__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator (
    _Safe_local_iterator< _Iterator, _Sequence > && __x ) [inline], [noexcept]
```

Move construction.

Postcondition

`__x` is singular and unattached

References [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach\(\)](#), [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=\(\)](#), and [std::swap\(\)](#).

_Safe_local_iterator() [5/5]

```
template<typename _Iterator , typename _Sequence >
template<typename _MutableIterator >
__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator (
    const _Safe_local_iterator< _MutableIterator, typename __gnu_cxx::__enable_if< _IsConstant::__value &&std::__are_same< _MutableIterator, _OtherIterator >::__value, _Sequence >::__value & __x ) [inline], [noexcept]
```

Converting constructor from a mutable iterator to a constant iterator.

References [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach\(\)](#).

6.158.3 Member Function Documentation**_M_attach()** [1/2]

```
template<typename _Iterator , typename _Sequence >
void __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach (
    _Safe_sequence_base * __seq ) [inline]
```

Attach iterator to the given sequence.

References [__gnu_debug::_Safe_local_iterator_base::_M_attach\(\)](#), and [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator\(\)](#).
Referenced by [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=\(\)](#), and [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=\(\)](#).

_M_attach() [2/2]

```
void __gnu_debug::_Safe_local_iterator_base::_M_attach (
    _Safe_sequence_base * __seq,
    bool __constant ) [protected], [inherited]
```

Attaches this iterator to the given container, detaching it from whatever container it was attached to originally. If the new container is the NULL pointer, the iterator is left unattached.

Referenced by [__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base\(\)](#), [__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base\(\)](#), and [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach\(\)](#).

`_M_attach_single()` [1/2]

```
template<typename _Iterator , typename _Sequence >
void __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach_single (
    _Safe_sequence_base * __seq ) [inline]
```

Likewise, but not thread-safe.

References `__gnu_debug::_Safe_local_iterator_base::_M_attach_single()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach_single()`.

`_M_attach_single()` [2/2]

```
void __gnu_debug::_Safe_local_iterator_base::_M_attach_single (
    _Safe_sequence_base * __seq,
    bool __constant ) throw ( ) [protected], [inherited]
```

Likewise, but not thread-safe.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach_single()`.

`_M_attached_to()`

```
bool __gnu_debug::_Safe_iterator_base::_M_attached_to (
    const _Safe_sequence_base * __seq ) const [inline], [inherited]
```

Determines if we are attached to the given sequence.

References `__gnu_debug::_Safe_iterator_base::_M_sequence`.

`_M_can_compare()`

```
bool __gnu_debug::_Safe_iterator_base::_M_can_compare (
    const _Safe_iterator_base & __x ) const throw ( ) [inherited]
```

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

`_M_dereferenceable()`

```
template<typename _Iterator , typename _Sequence >
bool __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_dereferenceable ( ) const
[inline]
```

Is the iterator dereferenceable?

References `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end()`, and `__gnu_debug::_Safe_iterator_base::_M_singular()`.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator*()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator[]()`.

`_M_detach()`

```
void __gnu_debug::_Safe_local_iterator_base::_M_detach ( ) [protected], [inherited]
```

Detach the iterator for whatever container it is attached to, if any.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::~operator=()`.

`_M_detach_single()`

```
void __gnu_debug::_Safe_local_iterator_base::_M_detach_single ( ) throw ( ) [protected], [inherited]
```

Likewise, but not thread-safe.

`_M_get_mutex()`

```
__gnu_cxx::mutex & __gnu_debug::_Safe_iterator_base::_M_get_mutex ( ) throw ( ) [protected],
[inherited]
```

For use in `_Safe_iterator`.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#).

_M_in_same_bucket()

```
template<typename _Iterator, typename _Sequence>
template<typename _Other>
bool __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_in_same_bucket (
    const _Safe_local_iterator<_Other, _Sequence> & __other) const [inline]
```

Is this iterator part of the same bucket as the other one?

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::bucket\(\)](#).

_M_incrementable()

```
template<typename _Iterator, typename _Sequence>
bool __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable () const [inline]
```

Is the iterator incrementable?

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end\(\)](#), and [__gnu_debug::_Safe_iterator_base::_M_singular\(\)](#).

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator--\(\)](#).

_M_invalidate()

```
void __gnu_debug::_Safe_iterator_base::_M_invalidate () [inline], [inherited]
```

Invalidate the iterator, making it singular.

References [__gnu_debug::_Safe_iterator_base::_M_version\(\)](#).

_M_is_begin()

```
template<typename _Iterator, typename _Sequence>
bool __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_begin () const [inline]
```

Is this iterator equal to the sequence's begin(bucket) iterator?

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::bucket\(\)](#).

_M_is_end()

```
template<typename _Iterator, typename _Sequence>
bool __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end () const [inline]
```

Is this iterator equal to the sequence's end(bucket) iterator?

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::bucket\(\)](#).

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable\(\)](#).

_M_reset()

```
void __gnu_debug::_Safe_iterator_base::_M_reset () throw () [inherited]
```

Reset all member variables.

_M_singular()

```
bool __gnu_debug::_Safe_iterator_base::_M_singular () const throw () [inherited]
```

Is this iterator singular?

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_incrementable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable\(\)](#).

`_M_unlink()`

```
void __gnu_debug::_Safe_iterator_base::_M_unlink ( ) throw ( ) [inline], [inherited]
```

Unlink itself

References [__gnu_debug::_Safe_iterator_base::_M_next](#), and [__gnu_debug::_Safe_iterator_base::_M_prior](#).

`_S_constant()`

```
template<typename _Iterator , typename _Sequence >
static constexpr bool __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_S_constant ( )
[inline], [static], [constexpr]
```

Determine if this is a constant iterator.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_detach\(\)](#).

`base()`

```
template<typename _Iterator , typename _Sequence >
_Iterator & __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base ( ) [inline], [noexcept]
```

Return the underlying iterator.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator*\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator--\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=\(\)](#).

`bucket()`

```
template<typename _Iterator , typename _Sequence >
size_type __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::bucket ( ) const [inline]
```

Return the bucket.

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_in_same_bucket\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator*\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end\(\)](#).

`operator _Iterator()`

```
template<typename _Iterator , typename _Sequence >
__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator _Iterator ( ) const [inline]
```

Conversion to underlying non-debug iterator to allow better interaction with non-debug containers.

`operator*()`

```
template<typename _Iterator , typename _Sequence >
reference __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator* ( ) const [inline]
```

Iterator dereference.

Precondition

iterator is dereferenceable

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator*\(\)](#).

`operator++()` [1/2]

```
template<typename _Iterator , typename _Sequence >
_Safe_local_iterator & __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++ ( )
[inline]
```

Iterator preincrement.

Precondition

iterator is incrementable

References [__gnu_debug::_Safe_iterator_base::_M_get_mutex\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_increment\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

operator++() [2/2]

```
template<typename _Iterator , typename _Sequence >
__Safe_local_iterator __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++ (
    int ) [inline]
```

Iterator postincrement.

Precondition

iterator is incrementable

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_increment\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_sequence\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

operator->()

```
template<typename _Iterator , typename _Sequence >
pointer __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator-> ( ) const [inline]
```

Iterator dereference.

Precondition

iterator is dereferenceable

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

operator=() [1/2]

```
template<typename _Iterator , typename _Sequence >
__Safe_local_iterator & __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator= (
    __Safe_local_iterator< _Iterator, _Sequence > && __x ) [inline], [noexcept]
```

Move assignment.

Postcondition

__x is singular and unattached

References [std::_addressof\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_sequence\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_version\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

operator=() [2/2]

```
template<typename _Iterator , typename _Sequence >
__Safe_local_iterator & __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator= (
    const __Safe_local_iterator< _Iterator, _Sequence > & __x ) [inline]
```

Copy assignment.

References [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_get_mutex\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_sequence\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_version\(\)](#), [__gnu_debug::_Safe_iterator_base::_M_version\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base\(\)](#).

6.158.4 Member Data Documentation

`_M_next`

`_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::_M_next` [inherited]

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Referenced by `__gnu_debug::Safe_sequence<_SafeSequence>::_M_invalidate_if()`, `__gnu_debug::Safe_sequence<_Sequence>` and `__gnu_debug::Safe_iterator_base::_M_unlink()`.

`_M_prior`

`_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::_M_prior` [inherited]

Pointer to the previous iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Referenced by `__gnu_debug::Safe_sequence<_SafeSequence>::_M_invalidate_if()`, `__gnu_debug::Safe_sequence<_Sequence>` and `__gnu_debug::Safe_iterator_base::_M_unlink()`.

`_M_sequence`

`_Safe_sequence_base* __gnu_debug::_Safe_iterator_base::_M_sequence` [inherited]

The sequence this iterator references; may be `NULL` to indicate a singular iterator.

Referenced by `__gnu_debug::Safe_iterator_base::_Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator_base::_Safe_local_iterator_base()`, `__gnu_debug::Safe_iterator_base::_M_attached_to()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator++()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator--()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`.

`_M_version`

`unsigned int __gnu_debug::_Safe_iterator_base::_M_version` [inherited]

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Referenced by `__gnu_debug::Safe_iterator_base::_M_invalidate()`, `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence, _Category>::operator=()`.

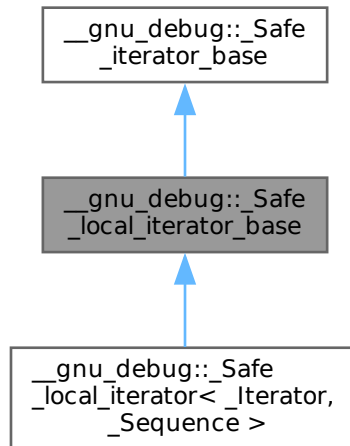
The documentation for this class was generated from the following files:

- `formatter.h`
- `safe_local_iterator.h`
- `safe_local_iterator.tcc`

6.159 `__gnu_debug::Safe_local_iterator_base` Class Reference

```
#include <safe_unordered_base.h>
```


Inheritance diagram for `__gnu_debug::_Safe_local_iterator_base`:



Public Member Functions

- `bool _M_attached_to (const _Safe_sequence_base * __seq) const`
- `bool _M_can_compare (const _Safe_iterator_base & __x) const throw ()`
- `void _M_invalidate ()`
- `void _M_reset () throw ()`
- `bool _M_singular () const throw ()`
- `void _M_unlink () throw ()`

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- `unsigned int _M_version`

Protected Member Functions

- `_Safe_local_iterator_base ()`
- `_Safe_local_iterator_base (const _Safe_local_iterator_base & __x, bool __constant)`
- `_Safe_local_iterator_base (const _Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach (_Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw ()`
- `void _M_detach ()`
- `void _M_detach_single () throw ()`
- `_Safe_unordered_container_base * _M_get_container () const noexcept`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`

6.159.1 Detailed Description

Basic functionality for a *safe* iterator.

The `_Safe_local_iterator_base` base class implements the functionality of a safe local iterator that is not specific to a particular iterator type. It contains a pointer back to the container it references along with iterator version information and pointers to form a doubly-linked list of local iterators referenced by the container.

This class must not perform any operations that can throw an exception, or the exception guarantees of derived iterators will be broken.

6.159.2 Constructor & Destructor Documentation

`_Safe_local_iterator_base()` [1/3]

```
__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base ( ) [inline], [protected]
```

Initializes the iterator and makes it singular.

`_Safe_local_iterator_base()` [2/3]

```
__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base (
    const \_Safe\_sequence\_base * __seq,
    bool __constant ) [inline], [protected]
```

Initialize the iterator to reference the container pointed to by `__seq`. `__constant` is true when we are initializing a constant local iterator, and false if it is a mutable local iterator. Note that `__seq` may be NULL, in which case the iterator will be singular. Otherwise, the iterator will reference `__seq` and be nonsingular.

References [_M_attach\(\)](#).

`_Safe_local_iterator_base()` [3/3]

```
__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base (
    const \_Safe\_local\_iterator\_base & __x,
    bool __constant ) [inline], [protected]
```

Initializes the iterator to reference the same container that `__x` does. `__constant` is true if this is a constant iterator, and false if it is mutable.

References [_M_attach\(\)](#), and [__gnu_debug::_Safe_iterator_base::_M_sequence](#).

6.159.3 Member Function Documentation

`_M_attach()`

```
void __gnu_debug::_Safe_local_iterator_base::_M_attach (
    \_Safe\_sequence\_base * __seq,
    bool __constant ) [protected]
```

Attaches this iterator to the given container, detaching it from whatever container it was attached to originally. If the new container is the NULL pointer, the iterator is left unattached.

Referenced by [_Safe_local_iterator_base\(\)](#), [_Safe_local_iterator_base\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence](#)

`_M_attach_single()`

```
void __gnu_debug::_Safe_local_iterator_base::_M_attach_single (
    \_Safe\_sequence\_base * __seq,
    bool __constant ) throw ( ) [protected]
```

Likewise, but not thread-safe.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach_single\(\)](#).

M_attached_to()

```
bool __gnu_debug::_Safe_iterator_base::_M_attached_to (
    const _Safe_sequence_base * __seq ) const [inline], [inherited]
```

Determines if we are attached to the given sequence.

References [__gnu_debug::_Safe_iterator_base::_M_sequence](#).

M_can_compare()

```
bool __gnu_debug::_Safe_iterator_base::_M_can_compare (
    const _Safe_iterator_base & __x ) const throw ( ) [inherited]
```

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

M_detach()

```
void __gnu_debug::_Safe_local_iterator_base::_M_detach ( ) [protected]
```

Detach the iterator for whatever container it is attached to, if any.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Itera](#)

M_detach_single()

```
void __gnu_debug::_Safe_local_iterator_base::_M_detach_single ( ) throw ( ) [protected]
```

Likewise, but not thread-safe.

M_get_mutex()

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_iterator_base::_M_get_mutex ( ) throw ( ) [protected],
[inherited]
```

For use in `_Safe_iterator`.

Referenced by [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Se](#),
[__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Cate](#),
[__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), [__gnu_debug::_Safe_local_iterator<_Iterator, _Sequen](#),
[__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _Se](#)

M_invalidate()

```
void __gnu_debug::_Safe_iterator_base::_M_invalidate ( ) [inline], [inherited]
```

Invalidate the iterator, making it singular.

References [__gnu_debug::_Safe_iterator_base::_M_version](#).

M_reset()

```
void __gnu_debug::_Safe_iterator_base::_M_reset ( ) throw ( ) [inherited]
```

Reset all member variables

M_singular()

```
bool __gnu_debug::_Safe_iterator_base::_M_singular ( ) const throw ( ) [inherited]
```

Is this iterator singular?

Referenced by [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_Safe_iterator\(\)](#), [__gnu_debug::_Safe_local_iterator<](#)
[__gnu_debug::_check_singular_aux\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::_M_dereferenceable\(\)](#),
[__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable\(\)](#), [__gnu_debug::_Safe_iterator<_Iterator, _Sequen](#),
[__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable\(\)](#), and [__gnu_debug::_Safe_local_iterator<_Iterator, _](#)

void `__gnu_debug::_Safe_iterator_base::_M_unlink` () throw () `[inline]`, `[inherited]`
 Unlink itself
 References `__gnu_debug::_Safe_iterator_base::_M_next`, and `__gnu_debug::_Safe_iterator_base::_M_prior`.

_M_next

_M_prior

_M_sequence

_M_version

unsigned int __gnu_debug::_Safe_iterator_base::_M_version [inherited]

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Referenced by `__gnu_debug::_Safe_iterator_base::_M_invalidate()`, `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_`
`__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`, `__gnu_debug::_Safe_iterator<_Iterator, _Sequence, _Category>::operator=()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

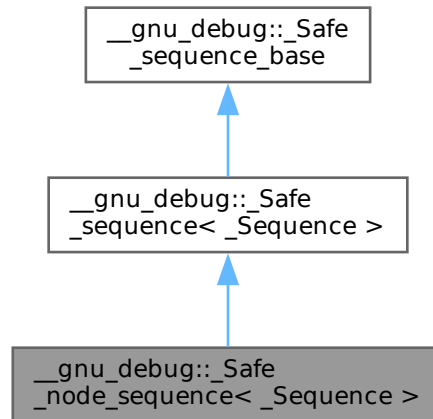
The documentation for this class was generated from the following file:

- `safe unordered base.h`

6.160 gnu_debug:: Safe_node_sequence<_Sequence > Class Template Reference

```
#include <safe_sequence.h>
```

Inheritance diagram for `__gnu_debug::_Safe_node_sequence<_Sequence>`:



Public Member Functions

- `template<typename _Predicate>`
`void _M_invalidate_if (_Predicate __pred)`
- `template<typename _Predicate>`
`void _M_transfer_from_if (_Safe_sequence &__from, _Predicate __pred)`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_invalidate_all () const`
- `void _M_revalidate_singular ()`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

6.160.1 Detailed Description

```

template<typename _Sequence>
class __gnu_debug::_Safe_node_sequence<_Sequence>

```

Like `_Safe_sequence` but with a special `_M_invalidate_all` implementation not invalidating past-the-end iterators. Used by node based sequence.

6.160.2 Member Function Documentation

`_M_detach_all()`

```
void __gnu_debug::_Safe_sequence_base::_M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

Referenced by `__gnu_debug::_Safe_sequence_base::~~Safe_sequence_base()`.

`_M_detach_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

`_M_get_mutex()`

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_sequence_base::_M_get_mutex ( ) throw ( ) [protected], [inherited]
```

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_invalidate_all()`

```
void __gnu_debug::_Safe_sequence_base::_M_invalidate_all ( ) const [inline], [protected], [inherited]
```

Invalidates all iterators.

References `__gnu_debug::_Safe_sequence_base::_M_version`.

`_M_invalidate_if()`

```
template<typename _Sequence>
template<typename _Predicate>
void __gnu_debug::_Safe_sequence<_Sequence>::_M_invalidate_if (
    _Predicate __pred ) [inherited]
```

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

`_M_revalidate_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ( ) [protected], [inherited]
```

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_swap()`

```
void __gnu_debug::_Safe_sequence_base::_M_swap (
    _Safe_sequence_base & __x ) [protected], [noexcept], [inherited]
```

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

`_M_transfer_from_if()`

```
template<typename _Sequence>
template<typename _Predicate>
```

```
void __gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if (
    __Safe_sequence< _Sequence > & __from,
    _Predicate __pred ) [inherited]
```

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

References [std::__addressof\(\)](#), [__gnu_debug::_Safe_sequence_base::_M_const_iterators](#), [__gnu_debug::_Safe_iterator_base::_M_detail](#), [__gnu_debug::_Safe_sequence_base::_M_get_mutex\(\)](#), [__gnu_debug::_Safe_sequence_base::_M_iterators](#), [__gnu_debug::_Safe_iterator_base::_M_prior](#), [__gnu_debug::_Safe_iterator_base::_M_sequence](#), and [__gnu_debug::_Safe_iterator_base](#).

6.160.3 Member Data Documentation

`__M_const_iterators`

[__Safe_iterator_base*](#) [__gnu_debug::_Safe_sequence_base::_M_const_iterators](#) [inherited]

The list of constant iterators that reference this container.

Referenced by [__gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if\(\)](#).

`__M_iterators`

[__Safe_iterator_base*](#) [__gnu_debug::_Safe_sequence_base::_M_iterators](#) [inherited]

The list of mutable iterators that reference this container.

Referenced by [__gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if\(\)](#).

`__M_version`

unsigned int [__gnu_debug::_Safe_sequence_base::_M_version](#) [mutable], [inherited]

The container version number. This number may never be 0.

Referenced by [__gnu_debug::_Safe_sequence_base::_M_invalidate_all\(\)](#), and [__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence](#).

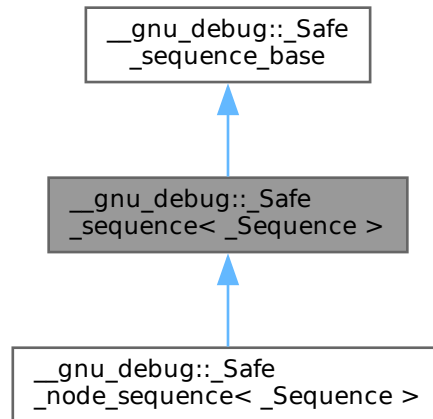
The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

6.161 `__gnu_debug::_Safe_sequence< _Sequence >` Class Template Reference

```
#include <safe_sequence.h>
```

Inheritance diagram for `__gnu_debug::_Safe_sequence<_Sequence>`:



Public Member Functions

- `template<typename _Predicate>`
`void _M_invalidate_if (_Predicate __pred)`
- `template<typename _Predicate>`
`void _M_transfer_from_if (_Safe_sequence &__from, _Predicate __pred)`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all () const`
- `void _M_revalidate_singular ()`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

6.161.1 Detailed Description

`template<typename _Sequence>`
class `__gnu_debug::_Safe_sequence<_Sequence>`

Base class for constructing a *safe* sequence type that tracks iterators that reference it.

The class template `_Safe_sequence` simplifies the construction of *safe* sequences that track the iterators that reference the sequence, so that the iterators are notified of changes in the sequence that may affect their operation, e.g., if the container invalidates its iterators or is destructed. This class template may only be used by deriving from it and passing

the name of the derived class as its template parameter via the curiously recurring template pattern. The derived class must have `iterator` and `const_iterator` types that are instantiations of class template `_Safe_iterator` for this sequence. Iterators will then be tracked automatically.

6.161.2 Member Function Documentation

`_M_detach_all()`

```
void __gnu_debug::_Safe_sequence_base::_M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

Referenced by [__gnu_debug::_Safe_sequence_base::~~_Safe_sequence_base\(\)](#).

`_M_detach_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

`_M_get_mutex()`

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_sequence_base::_M_get_mutex ( ) throw ( ) [protected],  
[inherited]
```

For use in `_Safe_sequence`.

Referenced by [__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#).

`_M_invalidate_all()`

```
void __gnu_debug::_Safe_sequence_base::_M_invalidate_all ( ) const [inline], [protected], [inherited]
```

Invalidates all iterators.

References [__gnu_debug::_Safe_sequence_base::_M_version](#).

`_M_invalidate_if()`

```
template<typename _Sequence >  
template<typename _Predicate >  
void __gnu_debug::_Safe_sequence<_Sequence>::_M_invalidate_if (  
    _Predicate __pred )
```

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

`_M_revalidate_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ( ) [protected], [inherited]
```

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_swap()`

```
void __gnu_debug::_Safe_sequence_base::_M_swap (  
    _Safe_sequence_base & __x ) [protected], [noexcept], [inherited]
```

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

`_M_transfer_from_if()`

```
template<typename _Sequence>
template<typename _Predicate>
void __gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if (
    _Safe_sequence<_Sequence> & __from,
    _Predicate __pred)
```

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

References `std::addressof()`, `__gnu_debug::Safe_sequence_base::M_const_iterators`, `__gnu_debug::Safe_iterator_base::M_detail`, `__gnu_debug::Safe_sequence_base::M_get_mutex()`, `__gnu_debug::Safe_sequence_base::M_iterators`, `__gnu_debug::Safe_iterator_base::M_prior`, `__gnu_debug::Safe_iterator_base::M_sequence`, and `__gnu_debug::Safe_iterator_base::M_unordered_container`.

6.161.3 Member Data Documentation**`_M_const_iterators`**

`Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_iterators`

`Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_version`

`unsigned int __gnu_debug::Safe_sequence_base::M_version` [mutable], [inherited]

The container version number. This number may never be 0.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator++()`.

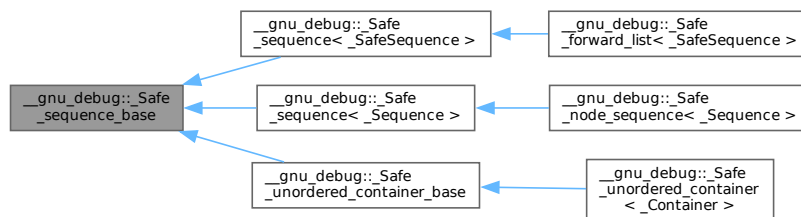
The documentation for this class was generated from the following files:

- `formatter.h`
- `safe_sequence.h`
- `safe_sequence.tcc`

6.162 `__gnu_debug::Safe_sequence_base` Class Reference

```
#include <safe_base.h>
```

Inheritance diagram for `__gnu_debug::Safe_sequence_base`:



Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- [_Safe_sequence_base](#) ([_Safe_sequence_base](#) &&__seq) noexcept
- [_Safe_sequence_base](#) (const [_Safe_sequence_base](#) &) noexcept
- [~_Safe_sequence_base](#) ()
- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_invalidate_all](#) () const
- void [_M_revalidate_singular](#) ()
- void [_M_swap](#) ([_Safe_sequence_base](#) &__x) noexcept

Friends

- class [_Safe_iterator_base](#)

6.162.1 Detailed Description

Base class that supports tracking of iterators that reference a sequence.

The [_Safe_sequence_base](#) class provides basic support for tracking iterators into a sequence. Sequences that track iterators must derived from [_Safe_sequence_base](#) publicly, so that safe iterators (which inherit [_Safe_iterator_base](#)) can attach to them. This class contains two linked lists of iterators, one for constant iterators and one for mutable iterators, and a version number that allows very fast invalidation of all iterators that reference the container.

This class must ensure that no operation on it may throw an exception, otherwise *safe* sequences may fail to provide the exception-safety guarantees required by the C++ standard.

6.162.2 Constructor & Destructor Documentation**[~_Safe_sequence_base\(\)](#)**

```
\_\_gnu\_debug::\_Safe\_sequence\_base::~\_Safe\_sequence\_base ( ) [inline], [protected]
```

Notify all iterators that reference this sequence that the sequence is being destroyed.

References [_M_detach_all\(\)](#).

6.162.3 Member Function Documentation**[_M_detach_all\(\)](#)**

```
void \_\_gnu\_debug::\_Safe\_sequence\_base::\_M\_detach\_all ( ) [protected]
```

Detach all iterators, leaving them singular.

Referenced by [~_Safe_sequence_base\(\)](#).

[_M_detach_singular\(\)](#)

```
void \_\_gnu\_debug::\_Safe\_sequence\_base::\_M\_detach\_singular ( ) [protected]
```

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->[_M_version](#) == [_M_version](#).

`_M_get_mutex()`

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_sequence_base::_M_get_mutex ( ) throw ( ) [protected]
```

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_invalidate_all()`

```
void __gnu_debug::_Safe_sequence_base::_M_invalidate_all ( ) const [inline], [protected]
```

Invalidates all iterators.

References `_M_version`.

`_M_revalidate_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ( ) [protected]
```

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_swap()`

```
void __gnu_debug::_Safe_sequence_base::_M_swap (
    __Safe_sequence_base & __x ) [protected], [noexcept]
```

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

6.162.4 Member Data Documentation**`_M_const_iterators`**

```
__Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_const_iterators
```

The list of constant iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_iterators`

```
__Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators
```

The list of mutable iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_version`

```
unsigned int __gnu_debug::_Safe_sequence_base::_M_version [mutable]
```

The container version number. This number may never be 0.

Referenced by `_M_invalidate_all()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

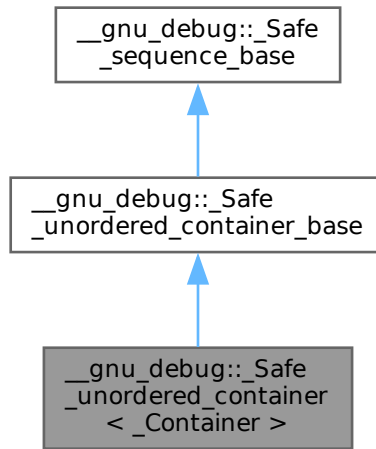
The documentation for this class was generated from the following file:

- `safe_base.h`

6.163 `__gnu_debug::_Safe_unordered_container<_Container>` Class Template Reference

```
#include <safe_unordered_container.h>
```

Inheritance diagram for `__gnu_debug::_Safe_unordered_container<_Container>`:



Public Member Functions

- `void _M_invalidate_all ()`
- `template<typename _Predicate>`
`void _M_invalidate_if (_Predicate __pred)`
- `template<typename _Predicate>`
`void _M_invalidate_local_if (_Predicate __pred)`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_const_local_iterators`
- `_Safe_iterator_base * _M_iterators`
- `_Safe_iterator_base * _M_local_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all () const`
- `void _M_invalidate_locals ()`
- `void _M_revalidate_singular ()`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`
- `void _M_swap (_Safe_unordered_container_base &__x) noexcept`

Static Protected Member Functions

- `template<typename _ExtractKey, typename _Source >`
`static _UContMergeGuard< _Source, _UContInvalidatePred< _ExtractKey, _Source > > _S_uc_guard (↵`
`_ExtractKey, _Source &__src)`
- `template<typename _ExtractKey, typename _Source >`
`static _UContMergeGuard< _Source, _UMContInvalidatePred< _ExtractKey, _Source > > _S_umc_guard (↵`
`_ExtractKey, _Source &__src)`

6.163.1 Detailed Description

`template<typename _Container>`
`class __gnu_debug::__Safe_unordered_container<_Container>`

Base class for constructing a *safe* unordered container type that tracks iterators that reference it.

The class template `_Safe_unordered_container` simplifies the construction of *safe* unordered containers that track the iterators that reference the container, so that the iterators are notified of changes in the container that may affect their operation, e.g., if the container invalidates its iterators or is destructed. This class template may only be used by deriving from it and passing the name of the derived class as its template parameter via the curiously recurring template pattern. The derived class must have `iterator` and `const_iterator` types that are instantiations of class template `↵ Safe_iterator` for this container and `local_iterator` and `const_local_iterator` types that are instantiations of class template `_Safe_local_iterator` for this container. Iterators will then be tracked automatically.

6.163.2 Member Function Documentation

`_M_detach_all()`

`void __gnu_debug::__Safe_unordered_container_base::_M_detach_all () [protected], [inherited]`
 Detach all iterators, leaving them singular.

`_M_detach_singular()`

`void __gnu_debug::__Safe_sequence_base::_M_detach_singular () [protected], [inherited]`
 Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

`_M_get_mutex()`

`__gnu_cxx::__mutex & __gnu_debug::__Safe_sequence_base::_M_get_mutex () throw () [protected], [inherited]`
 For use in `_Safe_sequence`.
 Referenced by `__gnu_debug::__Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_invalidate_all()`

`void __gnu_debug::__Safe_sequence_base::_M_invalidate_all () const [inline], [protected], [inherited]`
 Invalidates all iterators.
 References `__gnu_debug::__Safe_sequence_base::_M_version`.

`_M_invalidate_if()`

`template<typename _Container >`
`template<typename _Predicate >`

```
void __gnu_debug::_Safe_unordered_container< _Container >::_M_invalidate_if (
    _Predicate __pred )
```

Invalidates all iterators *x* that reference this container, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

`_M_invalidate_local_if()`

```
template<typename _Container >
template<typename _Predicate >
void __gnu_debug::_Safe_unordered_container< _Container >::_M_invalidate_local_if (
    _Predicate __pred )
```

Invalidates all local iterators *x* that reference this container, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal local iterators nested in the safe ones.

`_M_revalidate_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ( ) [protected], [inherited]
```

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_swap()` [1/2]

```
void __gnu_debug::_Safe_sequence_base::_M_swap (
    _Safe_sequence_base & __x ) [protected], [noexcept], [inherited]
```

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

`_M_swap()` [2/2]

```
void __gnu_debug::_Safe_unordered_container_base::_M_swap (
    _Safe_unordered_container_base & __x ) [protected], [noexcept], [inherited]
```

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

6.163.3 Member Data Documentation

`_M_const_iterators`

```
_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_const_iterators [inherited]
```

The list of constant iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if()`.

`_M_const_local_iterators`

```
_Safe_iterator_base* __gnu_debug::_Safe_unordered_container_base::_M_const_local_iterators [inherited]
```

The list of constant local iterators that reference this container.

`_M_iterators`

```
_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators [inherited]
```

The list of mutable iterators that reference this container.

Referenced by `__gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if()`.

`_M_local_iterators`

`_Safe_iterator_base*` `__gnu_debug::_Safe_unordered_container_base::_M_local_iterators` [inherited]

The list of mutable local iterators that reference this container.

`_M_version`

`unsigned int` `__gnu_debug::_Safe_sequence_base::_M_version` [mutable], [inherited]

The container version number. This number may never be 0.

Referenced by `__gnu_debug::_Safe_sequence_base::_M_invalidate_all()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence`

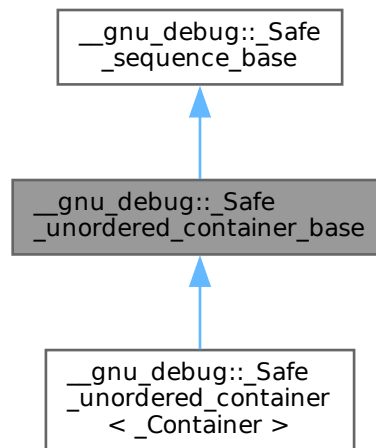
The documentation for this class was generated from the following files:

- [safe_unordered_container.h](#)
- [safe_unordered_container.tcc](#)

6.164 `__gnu_debug::_Safe_unordered_container_base` Class Reference

```
#include <safe_unordered_base.h>
```

Inheritance diagram for `__gnu_debug::_Safe_unordered_container_base`:

**Public Attributes**

- `_Safe_iterator_base` * `_M_const_iterators`
- `_Safe_iterator_base` * `_M_const_local_iterators`
- `_Safe_iterator_base` * `_M_iterators`
- `_Safe_iterator_base` * `_M_local_iterators`
- `unsigned int` `_M_version`

Protected Member Functions

- `_Safe_unordered_container_base` (`_Safe_unordered_container_base` && `_x`) noexcept
- `_Safe_unordered_container_base` (const `_Safe_unordered_container_base` &) noexcept

- `~_Safe_unordered_container_base () noexcept`
- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all () const`
- `void _M_revalidate_singular ()`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`
- `void _M_swap (_Safe_unordered_container_base &__x) noexcept`

Friends

- `class _Safe_local_iterator_base`

6.164.1 Detailed Description

Base class that supports tracking of local iterators that reference an unordered container.

The `_Safe_unordered_container_base` class provides basic support for tracking iterators into an unordered container. Containers that track iterators must derived from `_Safe_unordered_container_base` publicly, so that safe iterators (which inherit `_Safe_iterator_base`) can attach to them. This class contains four linked lists of iterators, one for constant iterators, one for mutable iterators, one for constant local iterators, one for mutable local iterators and a version number that allows very fast invalidation of all iterators that reference the container.

This class must ensure that no operation on it may throw an exception, otherwise *safe* containers may fail to provide the exception-safety guarantees required by the C++ standard.

6.164.2 Constructor & Destructor Documentation

`~_Safe_unordered_container_base()`

```
__gnu_debug::_Safe_unordered_container_base::~~_Safe_unordered_container_base ( ) [inline], [protected],
[noexcept]
```

Notify all iterators that reference this container that the container is being destroyed.

6.164.3 Member Function Documentation

`_M_detach_all()`

```
void __gnu_debug::_Safe_unordered_container_base::_M_detach_all ( ) [protected]
```

Detach all iterators, leaving them singular.

`_M_detach_singular()`

```
void __gnu_debug::_Safe_sequence_base::_M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, `i->_M_version == _M_version`.

`_M_get_mutex()`

```
__gnu_cxx::__mutex & __gnu_debug::_Safe_sequence_base::_M_get_mutex ( ) throw ( ) [protected],
[inherited]
```

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

`_M_invalidate_all()`

`void __gnu_debug::Safe_sequence_base::_M_invalidate_all () const [inline], [protected], [inherited]`
 Invalidates all iterators.

References [__gnu_debug::Safe_sequence_base::_M_version](#).

`_M_revalidate_singular()`

`void __gnu_debug::Safe_sequence_base::_M_revalidate_singular () [protected], [inherited]`
 Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

`_M_swap()` [1/2]

`void __gnu_debug::Safe_sequence_base::_M_swap (`
 [Safe_sequence_base](#) & __x) [protected], [noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

`_M_swap()` [2/2]

`void __gnu_debug::Safe_unordered_container_base::_M_swap (`
 [Safe_unordered_container_base](#) & __x) [protected], [noexcept]

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

6.164.4 Member Data Documentation**`_M_const_iterators`**

[Safe_iterator_base](#)* `__gnu_debug::Safe_sequence_base::_M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Referenced by [__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#).

`_M_const_local_iterators`

[Safe_iterator_base](#)* `__gnu_debug::Safe_unordered_container_base::_M_const_local_iterators`

The list of constant local iterators that reference this container.

`_M_iterators`

[Safe_iterator_base](#)* `__gnu_debug::Safe_sequence_base::_M_iterators` [inherited]

The list of mutable iterators that reference this container.

Referenced by [__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if\(\)](#).

`_M_local_iterators`

[Safe_iterator_base](#)* `__gnu_debug::Safe_unordered_container_base::_M_local_iterators`

The list of mutable local iterators that reference this container.

`_M_version`

`unsigned int __gnu_debug::Safe_sequence_base::_M_version` [mutable], [inherited]

The container version number. This number may never be 0.

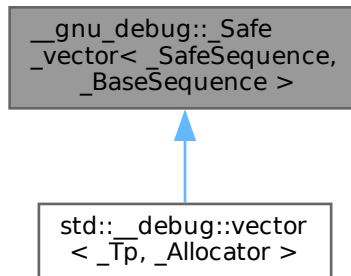
Referenced by [__gnu_debug::Safe_sequence_base::_M_invalidate_all\(\)](#), and [__gnu_debug::Safe_local_iterator<_Iterator, _Sequence](#)

The documentation for this class was generated from the following file:

- [safe_unordered_base.h](#)

6.165 `__gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence >` Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence >`:



Protected Member Functions

- `_Safe_vector` (`_Safe_vector` &&__x) noexcept
- `_Safe_vector` (const `_Safe_vector` &) noexcept
- `_Safe_vector` (size_type __n) noexcept
- `bool _M_requires_reallocation` (size_type __elements) const noexcept
- `void _M_update_guaranteed_capacity` () noexcept
- `_Safe_vector` & `operator=` (`_Safe_vector` &&__x) noexcept
- `_Safe_vector` & `operator=` (const `_Safe_vector` &) noexcept

Protected Attributes

- size_type `_M_guaranteed_capacity`

6.165.1 Detailed Description

```
template<typename _SafeSequence, typename _BaseSequence>
class __gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence >
```

Base class for Debug Mode vector.

Adds information about the guaranteed capacity, which is useful for detecting code which relies on non-portable implementation details of the libstdc++ reallocation policy.

The documentation for this class was generated from the following file:

- [debug/vector](#)

6.166 `__gnu_parallel::_SamplingSorter<__stable, _RAIter, _StrictWeakOrdering >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- void **operator()** (`_RAIter __first`, `_RAIter __last`, `_StrictWeakOrdering __comp`)

6.166.1 Detailed Description

```
template<bool __stable, class _RAIter, class _StrictWeakOrdering>
struct __gnu_parallel::__SamplingSorter< __stable, _RAIter, _StrictWeakOrdering >
```

Stable sorting functor.

Used to reduce code instantiation in `multiway_merge_sampling_splitting`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.167 `__gnu_parallel::__SamplingSorter< false, _RAIter, _StrictWeakOrdering >` Struct Template Reference

```
#include <multiway_merge.h>
```

Public Member Functions

- void **operator()** (`_RAIter __first`, `_RAIter __last`, `_StrictWeakOrdering __comp`)

6.167.1 Detailed Description

```
template<class _RAIter, class _StrictWeakOrdering>
struct __gnu_parallel::__SamplingSorter< false, _RAIter, _StrictWeakOrdering >
```

Non-`__stable` sorting functor.

Used to reduce code instantiation in `multiway_merge_sampling_splitting`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

6.168 `std::__detail::__Scanner< _CharT >` Class Template Reference

```
#include <regex_scanner.h>
```

Inherits `std::__detail::__ScannerBase`.

Public Types

- typedef const [std::ctype](#)< `_CharT` > `_CtypeT`
- typedef [regex_constants::syntax_option_type](#) `_FlagT`
- typedef [std::basic_string](#)< `_CharT` > `_StringT`
- enum `_TokenT` : unsigned {
`_S_token_anychar`, `_S_token_ord_char`, `_S_token_oct_num`, `_S_token_hex_num`,
`_S_token_backref`, `_S_token_subexpr_begin`, `_S_token_subexpr_no_group_begin`, `_S_token_↵`
`subexpr_lookahead_begin`,
`_S_token_subexpr_end`, `_S_token_bracket_begin`, `_S_token_bracket_neg_begin`, `_S_token_bracket_↵`
`end`,
`_S_token_interval_begin`, `_S_token_interval_end`, `_S_token_quoted_class`, `_S_token_char_class_name`
`,`
`_S_token_collsymbol`, `_S_token_equiv_class_name`, `_S_token_opt`, `_S_token_or`,
`_S_token_closure0`, `_S_token_closure1`, `_S_token_line_begin`, `_S_token_line_end`,
`_S_token_word_bound`, `_S_token_comma`, `_S_token_dup_count`, `_S_token_eof`,
`_S_token_bracket_dash`, `_S_token_unknown` }

Public Member Functions

- **_Scanner** (const _CharT * __begin, const _CharT * __end, [_FlagT](#) __flags, [std::locale](#) __loc)
- void **_M_advance** ()
- [_TokenT](#) **_M_get_token** () const noexcept
- const [_StringT](#) & **_M_get_value** () const noexcept

Protected Types

- enum **_StateT** { **_S_state_normal** , **_S_state_in_brace** , **_S_state_in_bracket** }

Protected Member Functions

- const char * **_M_find_escape** (char __c)
- bool **_M_is_awk** () const
- bool **_M_is_basic** () const
- bool **_M_is_ecma** () const
- bool **_M_is_extended** () const
- bool **_M_is_grep** () const

Protected Attributes

- bool **_M_at_bracket_start**
- const [std::pair](#)< char, char > **_M_awk_escape_tbl** [11]
- const char * **_M_basic_spec_char**
- const [std::pair](#)< char, char > **_M_ecma_escape_tbl** [8]
- const char * **_M_ecma_spec_char**
- const [std::pair](#)< char, char > * **_M_escape_tbl**
- const char * **_M_extended_spec_char**
- [_FlagT](#) **_M_flags**
- const char * **_M_spec_char**
- **_StateT** **_M_state**
- [_TokenT](#) **_M_token**
- const [std::pair](#)< char, [_TokenT](#) > **_M_token_tbl** [9]

6.168.1 Detailed Description

```
template<typename _CharT>
class std::__detail::_Scanner< _CharT >
```

Scans an input range for regex tokens.

The `_Scanner` class interprets the regular expression pattern in the input range passed to its constructor as a sequence of parse tokens passed to the regular expression compiler. The sequence of tokens provided depends on the flag settings passed to the constructor: different regular expression grammars will interpret the same input pattern in syntactically different ways.

6.168.2 Member Enumeration Documentation

_TokenT

```
enum std::__detail::_ScannerBase::_TokenT : unsigned [inherited]
```

Token types returned from the scanner.

The documentation for this class was generated from the following files:

- [regex_scanner.h](#)
- [regex_scanner.tcc](#)

6.169 `__gnu_debug::Sequence_traits<_Sequence>` Struct Template Reference

```
#include <safe_iterator.h>
```

Public Types

- typedef `_Distance_traits< typename _Sequence::iterator > _DistTraits`

Static Public Member Functions

- static `_DistTraits::_type _S_size` (const `_Sequence` &__seq)

6.169.1 Detailed Description

```
template<typename _Sequence>
```

```
struct __gnu_debug::Sequence_traits<_Sequence>
```

Sequence traits giving the size of a container if possible.

The documentation for this struct was generated from the following file:

- [safe_iterator.h](#)

6.170 `__gnu_parallel::Settings` Struct Reference

```
#include <settings.h>
```

Static Public Member Functions

- static const `_Settings` & `get` () throw ()
- static void `set` (`_Settings` &) throw ()

Public Attributes

- `_SequenceIndex` `accumulate_minimal_n`
- unsigned int `adjacent_difference_minimal_n`
- `_AlgorithmStrategy` `algorithm_strategy`
- unsigned int `cache_line_size`
- `_SequenceIndex` `count_minimal_n`
- `_SequenceIndex` `fill_minimal_n`
- `_FindAlgorithm` `find_algorithm`
- double `find_increasing_factor`
- `_SequenceIndex` `find_initial_block_size`
- `_SequenceIndex` `find_maximum_block_size`
- float `find_scale_factor`
- `_SequenceIndex` `find_sequential_search_size`
- `_SequenceIndex` `for_each_minimal_n`
- `_SequenceIndex` `generate_minimal_n`
- unsigned long long `L1_cache_size`
- unsigned long long `L2_cache_size`
- `_SequenceIndex` `max_element_minimal_n`
- `_SequenceIndex` `merge_minimal_n`
- unsigned int `merge_oversampling`
- `_SplittingAlgorithm` `merge_splitting`
- `_SequenceIndex` `min_element_minimal_n`

- [_MultiwayMergeAlgorithm](#) **multiway_merge_algorithm**
- int [multiway_merge_minimal_k](#)
- [_SequenceIndex](#) [multiway_merge_minimal_n](#)
- unsigned int [multiway_merge_oversampling](#)
- [_SplittingAlgorithm](#) **multiway_merge_splitting**
- [_SequenceIndex](#) [nth_element_minimal_n](#)
- [_SequenceIndex](#) [partial_sort_minimal_n](#)
- [_PartialSumAlgorithm](#) **partial_sum_algorithm**
- float [partial_sum_dilation](#)
- unsigned int [partial_sum_minimal_n](#)
- double [partition_chunk_share](#)
- [_SequenceIndex](#) [partition_chunk_size](#)
- [_SequenceIndex](#) [partition_minimal_n](#)
- [_SequenceIndex](#) [qsb_steals](#)
- unsigned int [random_shuffle_minimal_n](#)
- [_SequenceIndex](#) [replace_minimal_n](#)
- [_SequenceIndex](#) [search_minimal_n](#)
- [_SequenceIndex](#) [set_difference_minimal_n](#)
- [_SequenceIndex](#) [set_intersection_minimal_n](#)
- [_SequenceIndex](#) [set_symmetric_difference_minimal_n](#)
- [_SequenceIndex](#) [set_union_minimal_n](#)
- [_SortAlgorithm](#) **sort_algorithm**
- [_SequenceIndex](#) [sort_minimal_n](#)
- unsigned int [sort_mwms_oversampling](#)
- unsigned int [sort_qs_num_samples_preset](#)
- [_SequenceIndex](#) [sort_qsb_base_case_maximal_n](#)
- [_SplittingAlgorithm](#) **sort_splitting**
- unsigned int [TLB_size](#)
- [_SequenceIndex](#) [transform_minimal_n](#)
- [_SequenceIndex](#) [unique_copy_minimal_n](#)
- [_SequenceIndex](#) **workstealing_chunk_size**

6.170.1 Detailed Description

class `_Settings` Run-time settings for the parallel mode including all tunable parameters.

6.170.2 Member Function Documentation

get()

```
static const \_Settings & __gnu_parallel::_Settings::get ( ) throw ( ) [static]
```

Get the global settings.

Referenced by [__gnu_parallel::_find_template\(\)](#), [__gnu_parallel::_find_template\(\)](#), [__gnu_parallel::_find_template\(\)](#), [__gnu_parallel::_for_each_template_random_access_workstealing\(\)](#), [__gnu_parallel::_parallel_nth_element\(\)](#), [__gnu_parallel::_parallel_partial_sum\(\)](#), [__gnu_parallel::_parallel_partial_sum_linear\(\)](#), [__gnu_parallel::_parallel_partition\(\)](#), [__gnu_parallel::_parallel_random_shuffle_drs\(\)](#), [__gnu_parallel::_parallel_sort\(\)](#), [__gnu_parallel::_parallel_sort\(\)](#), [__gnu_parallel::_parallel_sort_qs_conquer\(\)](#), [__gnu_parallel::_qsb_local_sort_with_helping\(\)](#), [__gnu_parallel::_sequential_random_s](#), [__gnu_parallel::multiway_merge_sampling_splitting\(\)](#), [__gnu_parallel::parallel_multiway_merge\(\)](#), [__gnu_parallel::parallel_sort_mwms\(\)](#), and [__gnu_parallel::parallel_sort_mwms_pu\(\)](#).

set()

```
static void __gnu_parallel::_Settings::set (  
    _Settings & ) throw ( )    [static]
```

Set the global settings.

6.170.3 Member Data Documentation**accumulate_minimal_n**

[_SequenceIndex](#) `__gnu_parallel::_Settings::accumulate_minimal_n`

Minimal input size for accumulate.

adjacent_difference_minimal_n

`unsigned int __gnu_parallel::_Settings::adjacent_difference_minimal_n`

Minimal input size for adjacent_difference.

cache_line_size

`unsigned int __gnu_parallel::_Settings::cache_line_size`

Overestimation of cache line size. Used to avoid false sharing, i.e. elements of different threads are at least this amount apart.

Referenced by [__gnu_parallel::__for_each_template_random_access_workstealing\(\)](#).

count_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::count_minimal_n`

Minimal input size for count and count_if.

fill_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::fill_minimal_n`

Minimal input size for fill.

find_increasing_factor

`double __gnu_parallel::_Settings::find_increasing_factor`

Block size increase factor for find.

find_initial_block_size

[_SequenceIndex](#) `__gnu_parallel::_Settings::find_initial_block_size`

Initial block size for find.

Referenced by [__gnu_parallel::__find_template\(\)](#).

find_maximum_block_size

[_SequenceIndex](#) `__gnu_parallel::_Settings::find_maximum_block_size`

Maximal block size for find.

find_scale_factor

`float __gnu_parallel::_Settings::find_scale_factor`

Block size scale-down factor with respect to current position.

Referenced by [__gnu_parallel::__find_template\(\)](#).

find_sequential_search_size

[_SequenceIndex](#) `__gnu_parallel::_Settings::find_sequential_search_size`

Start with looking for this many elements sequentially, for find.

Referenced by [__gnu_parallel::__find_template\(\)](#), and [__gnu_parallel::__find_template\(\)](#).

for_each_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::for_each_minimal_n`

Minimal input size for for_each.

generate_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::generate_minimal_n`

Minimal input size for generate.

L1_cache_size

`unsigned long long __gnu_parallel::_Settings::L1_cache_size`

size of the L1 cache in bytes (underestimation).

L2_cache_size

`unsigned long long __gnu_parallel::_Settings::L2_cache_size`

size of the L2 cache in bytes (underestimation).

Referenced by [__gnu_parallel::__parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::__sequential_random_shuffle\(\)](#).

max_element_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::max_element_minimal_n`

Minimal input size for max_element.

merge_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::merge_minimal_n`

Minimal input size for merge.

merge_oversampling

`unsigned int __gnu_parallel::_Settings::merge_oversampling`

Oversampling factor for merge.

Referenced by [__gnu_parallel::multiway_merge_sampling_splitting\(\)](#), and [__gnu_parallel::parallel_multiway_merge\(\)](#).

min_element_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::min_element_minimal_n`

Minimal input size for min_element.

multiway_merge_minimal_k

`int __gnu_parallel::_Settings::multiway_merge_minimal_k`

Oversampling factor for multiway_merge.

multiway_merge_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::multiway_merge_minimal_n`

Minimal input size for multiway_merge.

multiway_merge_oversampling

`unsigned int __gnu_parallel::_Settings::multiway_merge_oversampling`
Oversampling factor for multiway_merge.

nth_element_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::nth_element_minimal_n`
Minimal input size for nth_element.
Referenced by [__gnu_parallel::__parallel_nth_element\(\)](#).

partial_sort_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::partial_sort_minimal_n`
Minimal input size for partial_sort.

partial_sum_dilation

`float __gnu_parallel::_Settings::partial_sum_dilation`
Ratio for partial_sum. Assume "sum and write result" to be this factor slower than just "sum".
Referenced by [__gnu_parallel::__parallel_partial_sum_linear\(\)](#).

partial_sum_minimal_n

`unsigned int __gnu_parallel::_Settings::partial_sum_minimal_n`
Minimal input size for partial_sum.

partition_chunk_share

`double __gnu_parallel::_Settings::partition_chunk_share`
Chunk size for partition, relative to input size. If > 0.0, this value overrides partition_chunk_size.
Referenced by [__gnu_parallel::__parallel_partition\(\)](#).

partition_chunk_size

[_SequenceIndex](#) `__gnu_parallel::_Settings::partition_chunk_size`
Chunk size for partition.
Referenced by [__gnu_parallel::__parallel_partition\(\)](#).

partition_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::partition_minimal_n`
Minimal input size for partition.
Referenced by [__gnu_parallel::__parallel_nth_element\(\)](#).

qsb_steals

[_SequenceIndex](#) `__gnu_parallel::_Settings::qsb_steals`
The number of stolen ranges in load-balanced quicksort.

random_shuffle_minimal_n

`unsigned int __gnu_parallel::_Settings::random_shuffle_minimal_n`
Minimal input size for random_shuffle.

replace_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::replace_minimal_n`
Minimal input size for `replace` and `replace_if`.

search_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::search_minimal_n`
Minimal input size for `search` and `search_n`.

set_difference_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::set_difference_minimal_n`
Minimal input size for `set_difference`.

set_intersection_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::set_intersection_minimal_n`
Minimal input size for `set_intersection`.

set_symmetric_difference_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::set_symmetric_difference_minimal_n`
Minimal input size for `set_symmetric_difference`.

set_union_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::set_union_minimal_n`
Minimal input size for `set_union`.

sort_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::sort_minimal_n`
Minimal input size for parallel sorting.

sort_mwms_oversampling

`unsigned int __gnu_parallel::_Settings::sort_mwms_oversampling`
Oversampling factor for `parallel_std::sort` (MWMS).
Referenced by [__gnu_parallel::parallel_sort_mwms\(\)](#), and [__gnu_parallel::parallel_sort_mwms_pu\(\)](#).

sort_qs_num_samples_preset

`unsigned int __gnu_parallel::_Settings::sort_qs_num_samples_preset`
Such many samples to take to find a good pivot (quicksort).

sort_qsb_base_case_maximal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::sort_qsb_base_case_maximal_n`
Maximal subsequence `__length` to switch to unbalanced `__base` case. Applies to `std::sort` with dynamically load-balanced quicksort.
Referenced by [__gnu_parallel::__qsb_local_sort_with_helping\(\)](#).

TLB_size

`unsigned int __gnu_parallel::_Settings::TLB_size`

size of the Translation Lookaside Buffer (underestimation).

Referenced by [__gnu_parallel::_parallel_random_shuffle_drs\(\)](#), and [__gnu_parallel::_sequential_random_shuffle\(\)](#).

transform_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::transform_minimal_n`

Minimal input size for parallel `std::transform`.

unique_copy_minimal_n

[_SequenceIndex](#) `__gnu_parallel::_Settings::unique_copy_minimal_n`

Minimal input size for `unique_copy`.

The documentation for this struct was generated from the following file:

- [settings.h](#)

6.171 `std::_Sp_ebo_helper<_Nm, _Tp, false >` Struct Template Reference

```
#include <shared_ptr_base.h>
```

Public Member Functions

- `_Sp_ebo_helper` (`_Tp` && `__tp`)
- `_Sp_ebo_helper` (`const _Tp` & `__tp`)

Static Public Member Functions

- `static _Tp & _S_get` (`_Sp_ebo_helper` & `__eboh`)

6.171.1 Detailed Description

```
template<int _Nm, typename _Tp>
```

```
struct std::_Sp_ebo_helper<_Nm, _Tp, false >
```

Specialization not using EBO.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

6.172 `std::_Sp_ebo_helper<_Nm, _Tp, true >` Struct Template Reference

```
#include <shared_ptr_base.h>
```

Inherits `_Tp`.

Public Member Functions

- `_Sp_ebo_helper` (`_Tp` && `__tp`)
- `_Sp_ebo_helper` (`const _Tp` & `__tp`)

Static Public Member Functions

- `static _Tp & _S_get` (`_Sp_ebo_helper` & `__eboh`)

6.172.1 Detailed Description

```
template<int _Nm, typename _Tp>
struct std::_Sp_ebo_helper<_Nm, _Tp, true >
```

Specialization using EBO.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

6.173 __gnu_parallel::_SplitConsistently< __exact, _RAlter, _Compare, _SortingPlacesIterator > Struct Template Reference

```
#include <multiway_mergesort.h>
```

6.173.1 Detailed Description

```
template<bool __exact, typename _RAlter, typename _Compare, typename _SortingPlacesIterator>
struct __gnu_parallel::_SplitConsistently< __exact, _RAlter, _Compare, _SortingPlacesIterator >
```

Split consistently.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

6.174 __gnu_parallel::_SplitConsistently< false, _RAlter, _Compare, _SortingPlacesIterator > Struct Template Reference

```
#include <multiway_mergesort.h>
```

Public Member Functions

- void **operator()** (const [_ThreadIndex](#) __iam, [_PMWMSSortingData](#)< _RAlter > *__sd, _Compare &__comp, const typename [std::iterator_traits](#)< _RAlter >::difference_type __num_samples) const

6.174.1 Detailed Description

```
template<typename _RAlter, typename _Compare, typename _SortingPlacesIterator>
struct __gnu_parallel::_SplitConsistently< false, _RAlter, _Compare, _SortingPlacesIterator >
```

Split by sampling.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

6.175 __gnu_parallel::_SplitConsistently< true, _RAlter, _Compare, _SortingPlacesIterator > Struct Template Reference

```
#include <multiway_mergesort.h>
```

Public Member Functions

- void **operator()** (const [_ThreadIndex](#) __iam, [_PMWMSSortingData](#)< _RAlter > *__sd, _Compare &__comp, const typename [std::iterator_traits](#)< _RAlter >::difference_type __num_samples) const

6.175.1 Detailed Description

```
template<typename _RAIter, typename _Compare, typename _SortingPlacesIterator>
struct __gnu_parallel::_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIterator >
```

Split by exact splitting.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

6.176 std::__detail::_StateSeq< _TraitsT > Class Template Reference

```
#include <regex_automaton.h>
```

Public Types

- typedef _NFA< _TraitsT > _RegexT

Public Member Functions

- _StateSeq (_RegexT &__nfa, _StateIdT __s)
- _StateSeq (_RegexT &__nfa, _StateIdT __s, _StateIdT __end)
- void _M_append (_StateIdT __id)
- void _M_append (const _StateSeq &__s)
- _StateSeq _M_clone ()

Public Attributes

- _StateIdT _M_end
- _RegexT & _M_nfa
- _StateIdT _M_start

6.176.1 Detailed Description

```
template<typename _TraitsT>
class std::__detail::_StateSeq< _TraitsT >
```

Describes a sequence of one or more _State, its current start and end(s). This structure contains fragments of an NFA during construction.

The documentation for this class was generated from the following files:

- [regex_automaton.h](#)
- [regex_automaton.tcc](#)

6.177 __gnu_cxx::_Std_pointer_impl< _Tp > Class Template Reference

```
#include <pointer.h>
```

Public Types

- typedef _Tp element_type

Public Member Functions

- _Tp * get () const
- bool operator< (const _Std_pointer_impl &__arg) const
- bool operator== (const _Std_pointer_impl &__arg) const
- void set (element_type *__arg)

6.177.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::Std_pointer_impl< _Tp >
```

A storage policy for use with `_Pointer_adapter<>` which yields a standard pointer.

A `_Storage_policy` is required to provide 4 things: 1) A `get()` API for returning the stored pointer value. 2) An `set()` API for storing a pointer value. 3) An `element_type` typedef to define the type this points to. 4) An `operator<()` to support pointer comparison. 5) An `operator==()` to support pointer comparison.

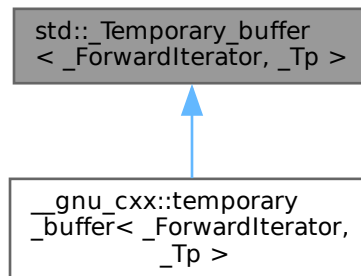
The documentation for this class was generated from the following file:

- [pointer.h](#)

6.178 `std::Temporary_buffer<_ForwardIterator, _Tp>` Class Template Reference

```
#include <stl_tempbuf.h>
```

Inheritance diagram for `std::Temporary_buffer<_ForwardIterator, _Tp>`:



Public Types

- typedef pointer **iterator**
- typedef value_type * **pointer**
- typedef ptrdiff_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- `_Temporary_buffer` (`_ForwardIterator __seed`, `size_type __original_len`)
- iterator `begin` ()
- iterator `end` ()
- `size_type requested_size` () const
- `size_type size` () const

Protected Attributes

- pointer **_M_buffer**
- `size_type _M_len`
- `size_type _M_original_len`

6.178.1 Detailed Description

```
template<typename _ForwardIterator, typename _Tp>
class std::_Temporary_buffer<_ForwardIterator, _Tp>
```

This class is used in two places: `stl_algo.h` and `ext/memory`, where it is wrapped as the `temporary_buffer` class. See `temporary_buffer` docs for more notes.

6.178.2 Constructor & Destructor Documentation

`_Temporary_buffer()`

```
template<typename _ForwardIterator, typename _Tp>
std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer (
    _ForwardIterator __seed,
    size_type __original_len )
```

Constructs a temporary buffer of a size somewhere between zero and the given length.

References `std::pair<_T1, _T2>::first`, and `std::pair<_T1, _T2>::second`.

6.178.3 Member Function Documentation

`begin()`

```
template<typename _ForwardIterator, typename _Tp>
iterator std::_Temporary_buffer<_ForwardIterator, _Tp>::begin ( ) [inline]
```

As per Table mumble.

`end()`

```
template<typename _ForwardIterator, typename _Tp>
iterator std::_Temporary_buffer<_ForwardIterator, _Tp>::end ( ) [inline]
```

As per Table mumble.

`requested_size()`

```
template<typename _ForwardIterator, typename _Tp>
size_type std::_Temporary_buffer<_ForwardIterator, _Tp>::requested_size ( ) const [inline]
```

Returns the size requested by the constructor; may be `>size()`.

`size()`

```
template<typename _ForwardIterator, typename _Tp>
size_type std::_Temporary_buffer<_ForwardIterator, _Tp>::size ( ) const [inline]
```

As per Table mumble.

The documentation for this class was generated from the following file:

- [stl_tempbuf.h](#)

6.179 `std::_Tuple_impl<_Idx, _Elements>` Struct Template Reference

6.179.1 Detailed Description

```
template<size_t _Idx, typename... _Elements>
struct std::_Tuple_impl<_Idx, _Elements>
```

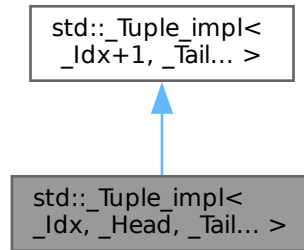
Contains the actual implementation of the `tuple` template, stored as a recursive inheritance hierarchy from the first element (most derived class) to the last (least derived class). The `Idx` parameter gives the 0-based index of the element stored at this point in the hierarchy; we use it to implement a constant-time `get()` operation.

The documentation for this struct was generated from the following file:

- [tuple](#)

6.180 std::_Tuple_impl< _Idx, _Head, _Tail... > Struct Template Reference

Inheritance diagram for std::_Tuple_impl< _Idx, _Head, _Tail... >:



Public Types

- typedef _Head_base< _Idx, _Head > **_Base**
- typedef [_Tuple_impl](#)< _Idx+1, _Tail... > **_Inherited**

Public Member Functions

- [_Tuple_impl](#) ([_Tuple_impl](#) &&)=default
- template<typename _UHead, typename... _UTails>
constexpr [_Tuple_impl](#) ([_Tuple_impl](#)< _Idx, _UHead, _UTails... > && __in)
- template<typename _UHead, typename... _UTail, typename = __enable_if_t<sizeof...(_Tail) == sizeof...(_UTail)>>
constexpr [_Tuple_impl](#) (_UHead && __head, _UTail &&... __tail)
- template<typename _Alloc >
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a)
- template<typename _Alloc >
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, [_Tuple_impl](#) && __in)
- template<typename _Alloc, typename _UHead, typename... _UTails>
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, [_Tuple_impl](#)< _Idx, _UHead, _UTails... > && __in)
- template<typename _Alloc, typename _UHead, typename... _UTail, typename = __enable_if_t<sizeof...(_Tail) == sizeof...(_UTail)>>
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, _UHead && __head, _UTail &&... __tail)
- template<typename _Alloc >
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, const _Head & __head, const _Tail &... __tail)
- template<typename _Alloc >
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, const [_Tuple_impl](#) & __in)
- template<typename _Alloc, typename _UHead, typename... _UTails>
constexpr [_Tuple_impl](#) (allocator_arg_t __tag, const _Alloc & __a, const [_Tuple_impl](#)< _Idx, _UHead, _UTails... > & __in)
- constexpr [_Tuple_impl](#) (const _Head & __head, const _Tail &... __tail)
- constexpr [_Tuple_impl](#) (const [_Tuple_impl](#) &)=default

- `template<typename... _UElements>`
`constexpr _Tuple_impl (const _Tuple_impl<_Idx, _UElements...> &__in)`
- `template<typename _UHead, typename... _UTails>`
`constexpr void _M_assign (_Tuple_impl<_Idx, _UHead, _UTails...> &&__in)`
- `template<typename... _UElements>`
`constexpr void _M_assign (const _Tuple_impl<_Idx, _UElements...> &__in)`
- `_Tuple_impl & operator= (const _Tuple_impl &)=delete`

Static Public Member Functions

- `static constexpr _Head & _M_head (_Tuple_impl &__t) noexcept`
- `static constexpr const _Head & _M_head (const _Tuple_impl &__t) noexcept`
- `static constexpr _Inherited & _M_tail (_Tuple_impl &__t) noexcept`
- `static constexpr const _Inherited & _M_tail (const _Tuple_impl &__t) noexcept`

Protected Member Functions

- `constexpr void _M_swap (_Tuple_impl &__in)`

6.180.1 Detailed Description

`template<size_t _Idx, typename _Head, typename... _Tail>`
`struct std::_Tuple_impl<_Idx, _Head, _Tail...>`

Recursive tuple implementation. Here we store the `Head` element and derive from a `Tuple_impl` containing the remaining elements (which contains the `Tail`).

The documentation for this struct was generated from the following file:

- [tuple](#)

6.181 `__gnu_cxx::Unqualified_type<_Tp>` Struct Template Reference

```
#include <pointer.h>
```

Public Types

- `typedef _Tp type`

6.181.1 Detailed Description

`template<typename _Tp>`
`struct __gnu_cxx::Unqualified_type<_Tp>`

This structure accommodates the way in which `std::iterator_traits<>` is normally specialized for `const T*`, so that `value_type` is still `T`.

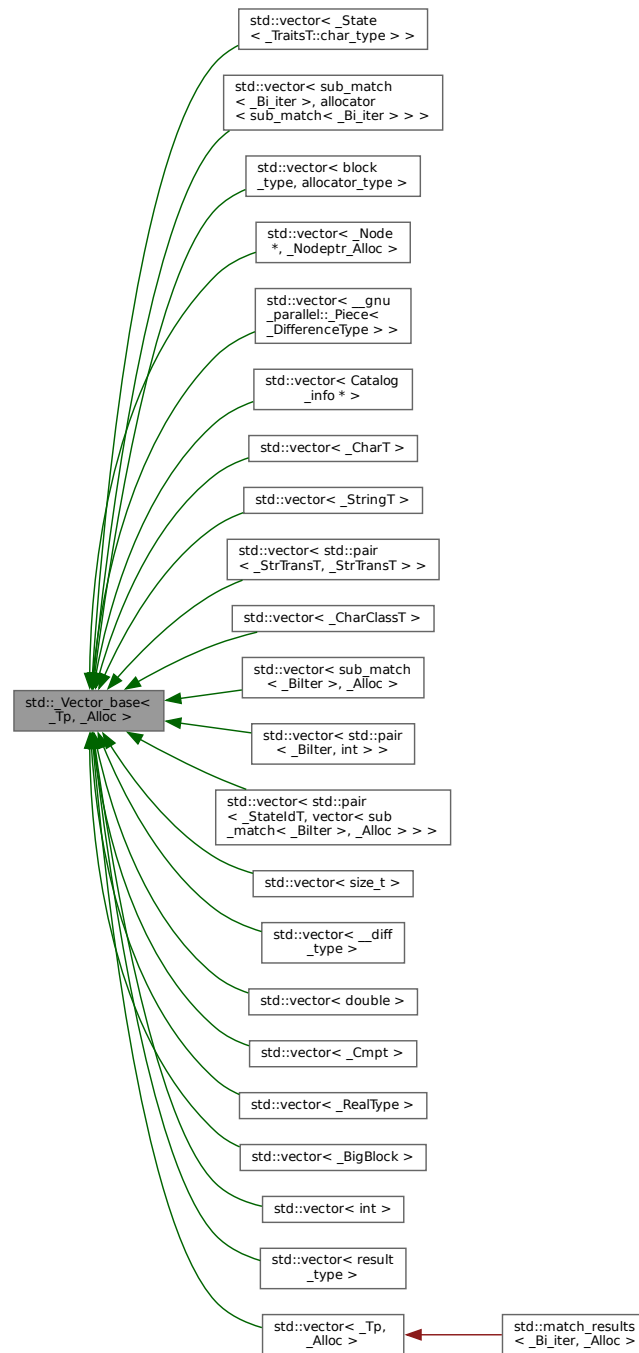
The documentation for this struct was generated from the following file:

- [pointer.h](#)

6.182 `std::_Vector_base<_Tp, _Alloc>` Struct Template Reference

```
#include <stl_vector.h>
```

Inheritance diagram for `std::_Vector_base<_Tp, _Alloc>`:



Public Types

- typedef `__gnu_cxx::__alloc_traits<_Alloc>::template rebind<_Tp>::other_Tp_alloc_type`
- typedef `_Alloc allocator_type`
- typedef `__gnu_cxx::__alloc_traits<_Tp_alloc_type>::pointer pointer`

Public Member Functions

- constexpr **_Vector_base** (_Tp_alloc_type &&__a) noexcept
- **_Vector_base** (_Vector_base &&)=default
- constexpr **_Vector_base** (_Vector_base &&__x, const allocator_type &__a)
- constexpr **_Vector_base** (const allocator_type &__a) noexcept
- constexpr **_Vector_base** (const allocator_type &__a, **_Vector_base** &&__x)
- constexpr **_Vector_base** (size_t __n)
- constexpr **_Vector_base** (size_t __n, const allocator_type &__a)
- constexpr pointer **_M_allocate** (size_t __n)
- constexpr void **_M_deallocate** (pointer __p, size_t __n)
- constexpr const _Tp_alloc_type & **_M_get_Tp_allocator** () const noexcept
- constexpr _Tp_alloc_type & **_M_get_Tp_allocator** () noexcept
- constexpr allocator_type **get_allocator** () const noexcept

Public Attributes

- _Vector_impl **_M_impl**

Protected Member Functions

- constexpr void **_M_create_storage** (size_t __n)

6.182.1 Detailed Description

template<typename _Tp, typename _Alloc>

struct std::_Vector_base< _Tp, _Alloc >

See bits/stl_deque.h's _Deque_base for an explanation.

The documentation for this struct was generated from the following file:

- [stl_vector.h](#)

6.183 std::add_const< _Tp > Struct Template Reference**Public Types**

- typedef _Tp const **type**

6.183.1 Detailed Description

template<typename _Tp>

struct std::add_const< _Tp >

add_const

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.184 std::add_cv< _Tp > Struct Template Reference**Public Types**

- typedef [add_const](#)< typename add_volatile< _Tp >::type >::type **type**

6.184.1 Detailed Description

```
template<typename _Tp>
struct std::add_cv<_Tp>
```

add_cv

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.185 std::add_lvalue_reference<_Tp> Struct Template Reference

Inherits std::__add_lvalue_reference_helper<_Tp, bool>.

Public Types

- typedef _Tp type

6.185.1 Detailed Description

```
template<typename _Tp>
struct std::add_lvalue_reference<_Tp>
```

add_lvalue_reference

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.186 std::add_pointer<_Tp> Struct Template Reference

Inherits std::__add_pointer_helper<_Tp, bool>.

Public Types

- typedef _Tp type

6.186.1 Detailed Description

```
template<typename _Tp>
struct std::add_pointer<_Tp>
```

add_pointer

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.187 std::add_rvalue_reference<_Tp> Struct Template Reference

Inherits std::__add_rvalue_reference_helper<_Tp, bool>.

Public Types

- typedef _Tp type

6.187.1 Detailed Description

```
template<typename _Tp>
struct std::add_rvalue_reference< _Tp >
```

add_rvalue_reference

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.188 std::add_volatile< _Tp > Struct Template Reference

Public Types

- typedef _Tp volatile **type**

6.188.1 Detailed Description

```
template<typename _Tp>
struct std::add_volatile< _Tp >
```

add_volatile

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.189 std::adopt_lock_t Struct Reference

```
#include <std_mutex.h>
```

6.189.1 Detailed Description

Assume the calling thread has already obtained mutex ownership and manage it.

The documentation for this struct was generated from the following file:

- [std_mutex.h](#)

6.190 std::aligned_storage< _Len, _Align > Struct Template Reference

6.190.1 Detailed Description

```
template<std::size_t _Len, std::size_t _Align = __alignof__(typename __aligned_storage_msa<_Len>::__↵
type)>
struct std::aligned_storage< _Len, _Align >
```

Alignment type.

The value of _Align is a default-alignment which shall be the most stringent alignment requirement for any C++ object type whose size is no greater than _Len (3.9). The member typedef type shall be a POD type suitable for use as uninitialized storage for any object whose size is at most _Len and whose alignment is a divisor of _Align.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.191 std::aligned_union< _Len, _Types > Struct Template Reference

Public Types

- typedef [aligned_storage](#)< _S_len, [alignment_value](#) >::type **type**

Static Public Attributes

- static const size_t [alignment_value](#)

6.191.1 Detailed Description

```
template<size_t _Len, typename... _Types>
struct std::aligned_union< _Len, _Types >
```

Provide aligned storage for types.

[meta.trans.other]

Provides aligned storage for any of the provided types of at least size _Len.

See also

[aligned_storage](#)

6.191.2 Member Typedef Documentation

type

```
template<size_t _Len, typename... _Types>
typedef aligned\_storage<_S_len,alignment\_value>::type std::aligned\_union< _Len, _Types >::type
```

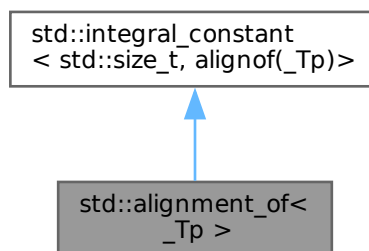
The storage.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.192 std::alignment_of< _Tp > Struct Template Reference

Inheritance diagram for std::alignment_of< _Tp >:



Public Types

- typedef [integral_constant](#)< std::size_t, __v > **type**
- typedef std::size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr std::size_t **value**

6.192.1 Detailed Description

```
template<typename _Tp>
struct std::alignment_of<_Tp>
```

alignment_of

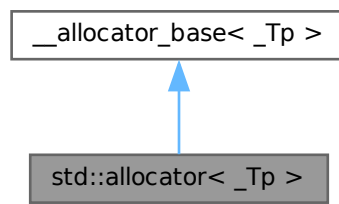
The documentation for this struct was generated from the following file:

- [type_traits](#)

6.193 std::allocator<_Tp> Class Template Reference

```
#include <allocator.h>
```

Inheritance diagram for std::allocator<_Tp>:



Public Types

- typedef ptrdiff_t **difference_type**
- using **is_always_equal** = [true_type](#)
- using **propagate_on_container_move_assignment** = [true_type](#)
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **allocator** (const [allocator](#) &__a) noexcept
- template<typename _Tp1 >
constexpr **allocator** (const [allocator](#)<_Tp1> &) noexcept
- constexpr _Tp * **allocate** (size_t __n)
- _Tp * **allocate** (size_type __n, const void * __p = static_cast<const void * >(0))
- constexpr void **deallocate** (_Tp * __p, size_t __n)
- [allocator](#) & **operator=** (const [allocator](#) &) = default

Friends

- constexpr bool **operator==** (const [allocator](#) &, const [allocator](#) &) noexcept

6.193.1 Detailed Description

```
template<typename _Tp>
class std::allocator<_Tp>
```

The *standard* allocator, as per C++03 [20.4.1].

See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/memory.html#std.util.memory.allocator> for further details.

Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

The documentation for this class was generated from the following file:

- [allocator.h](#)

6.194 std::allocator< void > Class Reference

```
#include <allocator.h>
```

Public Types

- typedef ptrdiff_t **difference_type**
- using **is_always_equal** = [true_type](#)
- using **propagate_on_container_move_assignment** = [true_type](#)
- typedef size_t **size_type**
- typedef void **value_type**

Public Member Functions

- template<typename _Up >
constexpr **allocator** (const [allocator](#)<_Up> &) noexcept

6.194.1 Detailed Description

`allocator<void>` specialization.

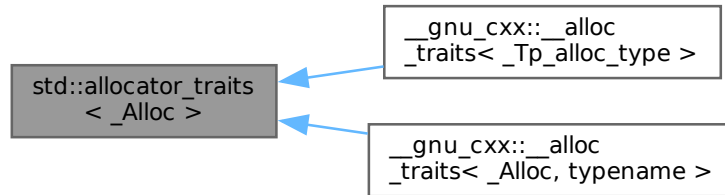
The documentation for this class was generated from the following file:

- [allocator.h](#)

6.195 std::allocator_traits< _Alloc > Struct Template Reference

```
#include <memory>
```

Inheritance diagram for std::allocator_traits<_Alloc>:



Public Types

- typedef `_Alloc` `allocator_type`
- using `const_pointer` = typename `_Ptr<__c_pointer, const value_type>::type`
- using `const_void_pointer` = typename `_Ptr<__cv_pointer, const void>::type`
- using `difference_type` = typename `_Diff<_Alloc, pointer>::type`
- using `is_always_equal` = `__detected_or_t<typename is_empty<_Alloc>::type, __equal, _Alloc>`
- using `pointer` = `__detected_or_t<value_type*, __pointer, _Alloc>`
- using `propagate_on_container_copy_assignment` = `__detected_or_t<false_type, __pocca, _Alloc>`
- using `propagate_on_container_move_assignment` = `__detected_or_t<false_type, __pocma, _Alloc>`
- using `propagate_on_container_swap` = `__detected_or_t<false_type, __pocs, _Alloc>`
- template<typename `_Tp`>
using `rebind_alloc` = `__alloc_rebind<_Alloc, _Tp>`
- template<typename `_Tp`>
using `rebind_traits` = `allocator_traits<rebind_alloc<_Tp>>`
- using `size_type` = typename `_Size<_Alloc, difference_type>::type`
- typedef `_Alloc::value_type` `value_type`
- using `void_pointer` = typename `_Ptr<__v_pointer, void>::type`

Static Public Member Functions

- static constexpr `pointer` `allocate` (`_Alloc` &`_a`, `size_type` `_n`)
- static constexpr `pointer` `allocate` (`_Alloc` &`_a`, `size_type` `_n`, `const_void_pointer` `_hint`)
- template<typename `_Tp`, typename... `_Args`>
static constexpr auto `construct` (`_Alloc` &`_a`, `_Tp` *`_p`, `_Args` &&... `_args`) noexcept(noexcept(`_S_construct`(`_a`, `_p`, `std::forward<_Args>(_args)...`))) -> `decltype(_S_construct(_a, _p, std::forward<_Args>(_args)...`)...)
- static constexpr void `deallocate` (`_Alloc` &`_a`, `pointer` `_p`, `size_type` `_n`)
- template<typename `_Tp`>
static constexpr void `destroy` (`_Alloc` &`_a`, `_Tp` *`_p`) noexcept(noexcept(`_S_destroy`(`_a`, `_p`, 0)))
- static constexpr `size_type` `max_size` (const `_Alloc` &`_a`) noexcept
- static constexpr `_Alloc` `select_on_container_copy_construction` (const `_Alloc` &`_rhs`)

6.195.1 Detailed Description

```
template<typename _Alloc>
struct std::allocator_traits< _Alloc >
```

Uniform interface to all allocator types.

Since

C++11

6.195.2 Member Typedef Documentation

allocator_type

```
template<typename _Alloc >
typedef _Alloc std::allocator_traits< _Alloc >::allocator_type
The allocator type.
```

const_pointer

```
template<typename _Alloc >
using std::allocator_traits< _Alloc >::const_pointer = typename _Ptr<__c_pointer, const value_type>↵
::type
The allocator's const pointer type.
Alloc::const_pointer if that type exists, otherwise pointer_traits<pointer>::rebind<const
value_type>
```

const_void_pointer

```
template<typename _Alloc >
using std::allocator_traits< _Alloc >::const_void_pointer = typename _Ptr<__cv_pointer, const
void>::type
The allocator's const void pointer type.
Alloc::const_void_pointer if that type exists, otherwise pointer_traits<pointer>::rebind<const
void>
```

difference_type

```
template<typename _Alloc >
using std::allocator_traits< _Alloc >::difference_type = typename _Diff<_Alloc, pointer>::type
The allocator's difference type.
Alloc::difference_type if that type exists, otherwise pointer_traits<pointer>::difference↵
_type
```

is_always_equal

```
template<typename _Alloc >
using std::allocator_traits< _Alloc >::is_always_equal = __detected_or_t<typename is_empty<↵
_Alloc>::type, __equal, _Alloc>
Whether all instances of the allocator type compare equal.
Alloc::is_always_equal if that type exists, otherwise is_empty<Alloc>::type
```

pointer

```
template<typename _Alloc >
using std::allocator_traits< _Alloc >::pointer = __detected_or_t<value_type*, __pointer, _Alloc>
```

The allocator's pointer type.

Alloc::pointer if that type exists, otherwise value_type*

propagate_on_container_copy_assignment

```
template<typename _Alloc>
using std::allocator_traits<_Alloc>::propagate_on_container_copy_assignment = __detected_or_t<false_type, __pocca, _Alloc>
```

How the allocator is propagated on copy assignment.

Alloc::propagate_on_container_copy_assignment if that type exists, otherwise false_type

propagate_on_container_move_assignment

```
template<typename _Alloc>
using std::allocator_traits<_Alloc>::propagate_on_container_move_assignment = __detected_or_t<false_type, __pocma, _Alloc>
```

How the allocator is propagated on move assignment.

Alloc::propagate_on_container_move_assignment if that type exists, otherwise false_type

propagate_on_container_swap

```
template<typename _Alloc>
using std::allocator_traits<_Alloc>::propagate_on_container_swap = __detected_or_t<false_type, __pocs, _Alloc>
```

How the allocator is propagated on swap.

Alloc::propagate_on_container_swap if that type exists, otherwise false_type

size_type

```
template<typename _Alloc>
using std::allocator_traits<_Alloc>::size_type = typename _Size<_Alloc, difference_type>::type
```

The allocator's size type.

Alloc::size_type if that type exists, otherwise make_unsigned<difference_type>::type

value_type

```
template<typename _Alloc>
typedef _Alloc::value_type std::allocator_traits<_Alloc>::value_type
```

The allocated type.

void_pointer

```
template<typename _Alloc>
using std::allocator_traits<_Alloc>::void_pointer = typename _Ptr<__v_pointer, void>::type
```

The allocator's void pointer type.

Alloc::void_pointer if that type exists, otherwise pointer_traits<pointer>::rebind<void>

6.195.3 Member Function Documentation

allocate() [1/2]

```
template<typename _Alloc>
static constexpr pointer std::allocator_traits<_Alloc>::allocate (
    _Alloc & __a,
    size_type __n ) [inline], [static], [constexpr]
```

Allocate memory.

Parameters

$_a$	An allocator.
$_n$	The number of objects to allocate space for.

Calls `a.allocate(n)`

allocate() [2/2]

```
template<typename _Alloc >
static constexpr pointer std::allocator_traits< _Alloc >::allocate (
    _Alloc & __a,
    size_type __n,
    const_void_pointer __hint ) [inline], [static], [constexpr]
```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for *n* objects of type `value_type`

Returns `a.allocate(n, hint)` if that expression is well-formed, otherwise returns `a.allocate(n)`

construct()

```
template<typename _Alloc >
template<typename _Tp , typename... _Args>
static constexpr auto std::allocator_traits< _Alloc >::construct (
    _Alloc & __a,
    _Tp * __p,
    _Args &&... __args ) -> decltype(_S_construct(__a, __p, std::forward<_Args>(__↵
args)...)) [inline], [static], [constexpr], [noexcept]
```

Construct an object of type `_Tp`

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for <code>Tp</code>
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<Args>(__args)...) if that expression is well-formed, otherwise uses placement-new to construct an object of type _Tp at location __p from the arguments __args...`

deallocate()

```
template<typename _Alloc >
```

```
static constexpr void std::allocator_traits<_Alloc>::deallocate (
    _Alloc & __a,
    pointer __p,
    size_type __n ) [inline], [static], [constexpr]
```

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

`destroy()`

```
template<typename _Alloc>
template<typename _Tp>
static constexpr void std::allocator_traits<_Alloc>::destroy (
    _Alloc & __a,
    _Tp * __p ) [inline], [static], [constexpr], [noexcept]
```

Destroy an object of type `_Tp`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Calls `__a.destroy(__p)` if that expression is well-formed, otherwise calls `__p->~_Tp()`

Referenced by [std::_Destroy\(\)](#).

`max_size()`

```
template<typename _Alloc>
static constexpr size_type std::allocator_traits<_Alloc>::max_size (
    const _Alloc & __a ) [inline], [static], [constexpr], [noexcept]
```

The maximum supported allocation size.

Parameters

<code>__a</code>	An allocator.
------------------	---------------

Returns

`__a.max_size()` or `numeric_limits<size_type>::max()`

Returns `__a.max_size()` if that expression is well-formed, otherwise returns `numeric_limits<size_type>::max()`

select_on_container_copy_construction()

```
template<typename _Alloc >
static constexpr _Alloc std::allocator_traits< _Alloc >::select_on_container_copy_construction (
    const _Alloc & __rhs ) [inline], [static], [constexpr]
```

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs.select_on_container_copy_construction()` or `__rhs`

Returns `__rhs.select_on_container_copy_construction()` if that expression is well-formed, otherwise returns `__rhs`

The documentation for this struct was generated from the following file:

- [bits/alloc_traits.h](#)

6.196 std::allocator_traits< allocator< _Tp > > Struct Template Reference

```
#include <alloc_traits.h>
```

Public Types

- using `allocator_type` = `allocator< _Tp >`
- using `const_pointer` = `const _Tp *`
- using `const_void_pointer` = `const void *`
- using `difference_type` = `std::ptrdiff_t`
- using `is_always_equal` = `true_type`
- using `pointer` = `_Tp *`
- using `propagate_on_container_copy_assignment` = `false_type`
- using `propagate_on_container_move_assignment` = `true_type`
- using `propagate_on_container_swap` = `false_type`
- template<typename _Up >
using `rebind_alloc` = `allocator< _Up >`
- template<typename _Up >
using `rebind_traits` = `allocator_traits< allocator< _Up > >`
- using `size_type` = `std::size_t`
- using `value_type` = `_Tp`
- using `void_pointer` = `void *`

Static Public Member Functions

- static constexpr `pointer allocate` (`allocator_type` &__a, `size_type` __n)
- static constexpr `pointer allocate` (`allocator_type` &__a, `size_type` __n, `const_void_pointer` __hint)
- template<typename _Up, typename... _Args>
static constexpr void `construct` (`allocator_type` &__a, _Up *__p, _Args &&... __args) noexcept(`std::is_nothrow_constructible< _Up, _Args... >::value`)
- static constexpr void `deallocate` (`allocator_type` &__a, `pointer` __p, `size_type` __n)
- template<typename _Up >
static constexpr void `destroy` (`allocator_type` &__a, _Up *__p) noexcept(`is_nothrow_destructible< _Up >::value`)
- static constexpr `size_type max_size` (`const allocator_type` &__a) noexcept
- static constexpr `allocator_type select_on_container_copy_construction` (`const allocator_type` &__rhs)

6.196.1 Detailed Description

```
template<typename _Tp>
struct std::allocator_traits< allocator< _Tp > >
```

Partial specialization for std::allocator.

6.196.2 Member Typedef Documentation

allocator_type

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::allocator_type = allocator<_Tp>
The allocator type.
```

const_pointer

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::const_pointer = const _Tp*
The allocator's const pointer type.
```

const_void_pointer

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::const_void_pointer = const void*
The allocator's const void pointer type.
```

difference_type

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::difference_type = std::ptrdiff_t
The allocator's difference type.
```

is_always_equal

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::is_always_equal = true_type
Whether all instances of the allocator type compare equal.
```

pointer

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::pointer = _Tp*
The allocator's pointer type.
```

propagate_on_container_copy_assignment

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::propagate_on_container_copy_assignment = false_type
How the allocator is propagated on copy assignment.
```

propagate_on_container_move_assignment

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::propagate_on_container_move_assignment = true_type
How the allocator is propagated on move assignment.
```


propagate_on_container_swap

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::propagate_on_container_swap = false_type
```

How the allocator is propagated on swap.

size_type

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::size_type = std::size_t
```

The allocator's size type.

value_type

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::value_type = _Tp
```

The allocated type.

void_pointer

```
template<typename _Tp >
using std::allocator_traits< allocator< _Tp > >::void_pointer = void*
```

The allocator's void pointer type.

6.196.3 Member Function Documentation**allocate() [1/2]**

```
template<typename _Tp >
static constexpr pointer std::allocator_traits< allocator< _Tp > >::allocate (
    allocator_type & __a,
    size_type __n ) [inline], [static], [constexpr]
```

Allocate memory.

Parameters

\leftarrow __a	An allocator.
\leftarrow __n	The number of objects to allocate space for.

Calls `a.allocate(n)`

allocate() [2/2]

```
template<typename _Tp >
static constexpr pointer std::allocator_traits< allocator< _Tp > >::allocate (
    allocator_type & __a,
    size_type __n,
    const_void_pointer __hint ) [inline], [static], [constexpr]
```

Allocate memory.

Parameters

__a	An allocator.
__n	The number of objects to allocate space for.

Parameters

<code>__hint</code>	Aid to locality.
---------------------	------------------

Returns

Memory of suitable size and alignment for n objects of type `value_type`

Returns `a.allocate(n, hint)`

construct()

```
template<typename _Tp >
template<typename _Up, typename... _Args>
static constexpr void std::allocator_traits< allocator< _Tp > >::construct (
    allocator_type & __a,
    _Up * __p,
    _Args &&... __args ) [inline], [static], [constexpr], [noexcept]
```

Construct an object of type `_Up`

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for an object of type <code>_Up</code> .
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<_Args>(__args)...) in C++11, C++14 and C++17. Changed in C++20 to call std::construct_at(__p, std::forward<_Args>(__args)...) instead.`

deallocate()

```
template<typename _Tp >
static constexpr void std::allocator_traits< allocator< _Tp > >::deallocate (
    allocator_type & __a,
    pointer __p,
    size_type __n ) [inline], [static], [constexpr]
```

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

destroy()

```
template<typename _Tp >
template<typename _Up >
```

```
static constexpr void std::allocator_traits< allocator< _Tp > >::destroy (
    allocator_type & __a,
    _Up * __p ) [inline], [static], [constexpr], [noexcept]
```

Destroy an object of type `_Up`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Calls `__a.destroy(__p)`.

`max_size()`

```
template<typename _Tp >
static constexpr size_type std::allocator_traits< allocator< _Tp > >::max_size (
    const allocator_type & __a ) [inline], [static], [constexpr], [noexcept]
```

The maximum supported allocation size.

Parameters

<code>__a</code>	An allocator.
------------------	---------------

Returns

`__a.max_size()`

`select_on_container_copy_construction()`

```
template<typename _Tp >
static constexpr allocator_type std::allocator_traits< allocator< _Tp > >::select_on_container←
_copy_construction (
    const allocator_type & __rhs ) [inline], [static], [constexpr]
```

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs`

The documentation for this struct was generated from the following file:

- [bits/alloc_traits.h](#)

6.197 `std::allocator_traits< allocator< void > >` Struct Reference

```
#include <alloc_traits.h>
```

Public Types

- using `allocator_type` = `allocator< void >`
- using `const_pointer` = `const void *`
- using `const_void_pointer` = `const void *`
- using `difference_type` = `std::ptrdiff_t`
- using `is_always_equal` = `true_type`
- using `pointer` = `void *`
- using `propagate_on_container_copy_assignment` = `false_type`
- using `propagate_on_container_move_assignment` = `true_type`
- using `propagate_on_container_swap` = `false_type`
- template<typename `_Up` >
using `rebind_alloc` = `allocator< _Up >`
- template<typename `_Up` >
using `rebind_traits` = `allocator_traits< allocator< _Up > >`
- using `size_type` = `std::size_t`
- using `value_type` = `void`
- using `void_pointer` = `void *`

Static Public Member Functions

- static void * `allocate` (`allocator_type` &, `size_type`, const void * = nullptr) = delete
- template<typename `_Up`, typename... `_Args` >
static constexpr void `construct` (`allocator_type` &, `_Up` * __p, `_Args` &&... __args) noexcept(`std::is_nothrow_constructible< _Up, _Args... >::value`)
- static void `deallocate` (`allocator_type` &, void *, `size_type`) = delete
- template<typename `_Up` >
static constexpr void `destroy` (`allocator_type` &, `_Up` * __p) noexcept(`is_nothrow_destructible< _Up >::value`)
- static `size_type` `max_size` (const `allocator_type` &) = delete
- static constexpr `allocator_type` `select_on_container_copy_construction` (const `allocator_type` & __rhs)

6.197.1 Detailed Description

Explicit specialization for `std::allocator<void>`.

6.197.2 Member Typedef Documentation

`allocator_type`

```
using std::allocator_traits< allocator< void > >::allocator_type = allocator<void>
```

The allocator type.

`const_pointer`

```
using std::allocator_traits< allocator< void > >::const_pointer = const void*
```

The allocator's const pointer type.

`const_void_pointer`

```
using std::allocator_traits< allocator< void > >::const_void_pointer = const void*
```

The allocator's const void pointer type.

difference_type

```
using std::allocator_traits< allocator< void > >::difference_type = std::ptrdiff_t
```

The allocator's difference type.

is_always_equal

```
using std::allocator_traits< allocator< void > >::is_always_equal = true_type
```

Whether all instances of the allocator type compare equal.

pointer

```
using std::allocator_traits< allocator< void > >::pointer = void*
```

The allocator's pointer type.

propagate_on_container_copy_assignment

```
using std::allocator_traits< allocator< void > >::propagate_on_container_copy_assignment = false_type
```

How the allocator is propagated on copy assignment.

propagate_on_container_move_assignment

```
using std::allocator_traits< allocator< void > >::propagate_on_container_move_assignment = true_type
```

How the allocator is propagated on move assignment.

propagate_on_container_swap

```
using std::allocator_traits< allocator< void > >::propagate_on_container_swap = false_type
```

How the allocator is propagated on swap.

size_type

```
using std::allocator_traits< allocator< void > >::size_type = std::size_t
```

The allocator's size type.

value_type

```
using std::allocator_traits< allocator< void > >::value_type = void
```

The allocated type.

void_pointer

```
using std::allocator_traits< allocator< void > >::void_pointer = void*
```

The allocator's void pointer type.

6.197.3 Member Function Documentation**allocate()**

```
static void * std::allocator_traits< allocator< void > >::allocate (
    allocator_type & ,
    size_type ,
    const void * = nullptr ) [static], [delete]
```

allocate is ill-formed for allocator<void>

construct()

```
template<typename _Up , typename... _Args>
static constexpr void std::allocator_traits< allocator< void > >::construct (
    allocator_type & ,
    _Up * __p,
    _Args &&... __args ) [inline], [static], [constexpr], [noexcept]
```

Construct an object of type `_Up`

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for an object of type <code>_Up</code> .
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<_Args>(__args)...) in C++11, C++14 and C++17. Changed in C++20 to call std::construct_at(__p, std::forward<_Args>(__args)...) instead. References std::_Construct\(\).`

deallocate()

```
static void std::allocator_traits< allocator< void > >::deallocate (
    allocator_type & ,
    void * ,
    size_type ) [static], [delete]
```

deallocate is ill-formed for `allocator<void>`

destroy()

```
template<typename _Up >
static constexpr void std::allocator_traits< allocator< void > >::destroy (
    allocator_type & ,
    _Up * __p ) [inline], [static], [constexpr], [noexcept]
```

Destroy an object of type `_Up`

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Invokes the destructor for `*__p`.

References [std::_Destroy\(\)](#).

max_size()

```
static size_type std::allocator_traits< allocator< void > >::max_size (
    const allocator_type & ) [static], [delete]
```

max_size is ill-formed for `allocator<void>`

select_on_container_copy_construction()

```
static constexpr allocator_type std::allocator_traits< allocator< void > >::select_on_container↵
```

```
_copy_construction (
    const allocator_type & __rhs ) [inline], [static], [constexpr]
```

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs`

The documentation for this struct was generated from the following file:

- bits/alloc_traits.h

6.198 `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >` Struct Template Reference

Public Types

- using `allocator_type` = `pmr::polymorphic_allocator< _Tp >`
- using `const_pointer` = `const _Tp *`
- using `const_void_pointer` = `const void *`
- using `difference_type` = `std::ptrdiff_t`
- using `is_always_equal` = `false_type`
- using `pointer` = `_Tp *`
- template<typename `_Up` >
using `rebind_alloc` = `pmr::polymorphic_allocator< _Up >`
- template<typename `_Up` >
using `rebind_traits` = `allocator_traits< pmr::polymorphic_allocator< _Up > >`
- using `size_type` = `std::size_t`
- using `value_type` = `_Tp`
- using `void_pointer` = `void *`

Static Public Member Functions

- static `pointer allocate` (`allocator_type &__a`, `size_type __n`)
- static `pointer allocate` (`allocator_type &__a`, `size_type __n`, `const_void_pointer`)
- template<typename `_Up`, typename... `_Args`>
static void `construct` (`allocator_type &__a`, `_Up *__p`, `_Args &&... __args`)
- static void `deallocate` (`allocator_type &__a`, `pointer __p`, `size_type __n`)
- template<typename `_Up` >
static constexpr void `destroy` (`allocator_type &`, `_Up *__p`) noexcept(`is_nothrow_destructible< _Up >::value`)
- static constexpr `size_type max_size` (`const allocator_type &`) noexcept
- using `propagate_on_container_copy_assignment` = `false_type`
- using `propagate_on_container_move_assignment` = `false_type`
- using `propagate_on_container_swap` = `false_type`
- static `allocator_type select_on_container_copy_construction` (`const allocator_type &`) noexcept

6.198.1 Detailed Description

```
template<typename _Tp>
struct std::allocator_traits< pmr::polymorphic_allocator< _Tp > >
```

Partial specialization for `std::pmr::polymorphic_allocator`.

6.198.2 Member Typedef Documentation

allocator_type

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::allocator_type = pmr::polymorphic_allocator<_Tp>
```

The allocator type.

const_pointer

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::const_pointer = const _Tp*
```

The allocator's const pointer type.

const_void_pointer

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::const_void_pointer = const
void*
```

The allocator's const void pointer type.

difference_type

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::difference_type = std::ptrdiff_t
```

The allocator's difference type.

is_always_equal

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::is_always_equal = false_type
```

Whether all instances of the allocator type compare equal.

pointer

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::pointer = _Tp*
```

The allocator's pointer type.

propagate_on_container_copy_assignment

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::propagate_on_container_copy_assignment = false_type
```

A `polymorphic_allocator` does not propagate when a container is copied, moved, or swapped.

propagate_on_container_move_assignment

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::propagate_on_container_move_assignment = false_type
```

A `polymorphic_allocator` does not propagate when a container is copied, moved, or swapped.

propagate_on_container_swap

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::propagate_on_container_swap =
false_type
```

A `polymorphic_allocator` does not propagate when a container is copied, moved, or swapped.

size_type

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::size_type = std::size_t
```

The allocator's size type.

value_type

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::value_type = _Tp
```

The allocated type.

void_pointer

```
template<typename _Tp >
using std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::void_pointer = void*
```

The allocator's void pointer type.

6.198.3 Member Function Documentation**allocate() [1/2]**

```
template<typename _Tp >
static pointer std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::allocate (
    allocator_type & __a,
    size_type __n ) [inline], [static]
```

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.

Calls `a.allocate(n)`.

allocate() [2/2]

```
template<typename _Tp >
static pointer std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::allocate (
    allocator_type & __a,
    size_type __n,
    const_void_pointer ) [inline], [static]
```

Allocate memory.

Parameters

\leftrightarrow _a	An allocator.
\leftrightarrow _n	The number of objects to allocate space for.

Returns

Memory of suitable size and alignment for n objects of type value_type.

The third parameter is ignored..

Returns a.allocate(n).

construct()

```
template<typename _Tp >
template<typename _Up , typename... _Args>
static void std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::construct (
    allocator_type & __a,
    _Up * __p,
    _Args &&... __args ) [inline], [static]
```

Construct an object of type _Up

Parameters

__a	An allocator.
__p	Pointer to memory of suitable size and alignment for an object of type _Up.
__args	Constructor arguments.

Calls __a.construct(__p, std::forward<_Args>(__args)...) in C++11, C++14 and C++17.
 Changed in C++20 to call std::construct_at(__p, std::forward<_Args>(__args)...) instead.

deallocate()

```
template<typename _Tp >
static void std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::deallocate (
    allocator_type & __a,
    pointer __p,
    size_type __n ) [inline], [static]
```

Deallocate memory.

Parameters

\leftrightarrow _a	An allocator.
\leftrightarrow _p	Pointer to the memory to deallocate.
\leftrightarrow _n	The number of objects space was allocated for.

Calls a.deallocate(p, n).

destroy()

```
template<typename _Tp >
template<typename _Up >
static constexpr void std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::destroy (
    allocator_type & ,
    _Up * __p ) [inline], [static], [constexpr], [noexcept]
```

Destroy an object of type `_Up`

Parameters

<code>↔ _a</code>	An allocator.
<code>↔ _p</code>	Pointer to the object to destroy

Calls `p->_Up()`.

max_size()

```
template<typename _Tp >
static constexpr size_type std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::max_size
(
    const allocator_type & ) [inline], [static], [constexpr], [noexcept]
```

The maximum supported allocation size.

Returns

```
numeric_limits<size_t>::max() / sizeof(value_type)
```

select_on_container_copy_construction()

```
template<typename _Tp >
static allocator_type std::allocator_traits< pmr::polymorphic_allocator< _Tp > >::select_on_↔
container_copy_construction (
    const allocator_type & ) [inline], [static], [noexcept]
```

A `polymorphic_allocator` does not propagate when a container is copied, moved, or swapped.

The documentation for this struct was generated from the following file:

- [memory_resource](#)

6.199 __gnu_cxx::limit_condition::always_adjustor Struct Reference

```
#include <throw_allocator.h>
```

Inherits `__gnu_cxx::limit_condition::adjustor_base`.

6.199.1 Detailed Description

Always enter the condition.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.200 __gnu_cxx::random_condition::always_adjustor Struct Reference

```
#include <throw_allocator.h>
```

Inherits `__gnu_cxx::random_condition::adjustor_base`.

6.200.1 Detailed Description

Always enter the condition.

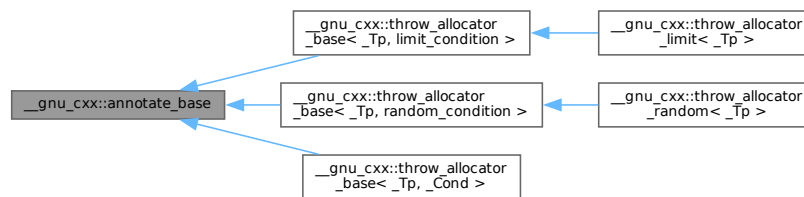
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.201 __gnu_cxx::annotate_base Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for __gnu_cxx::annotate_base:



Public Member Functions

- void **check** (size_t label)
- map_alloc_type::iterator **check_allocated** (void *p, size_t size)
- void **check_constructed** (size_t label)
- map_construct_type::iterator **check_constructed** (void *p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)

Static Public Member Functions

- static void **check** ()
- static size_t **get_label** ()
- static void **set_label** (size_t l)

Friends

- [std::ostream](#) & **operator<<** ([std::ostream](#) &, const [annotate_base](#) &)

6.201.1 Detailed Description

Base class for checking address and label information about allocations. Create a std::map between the allocated address (void*) and a datum for annotations, which are a pair of numbers corresponding to label and allocated size.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.202 std::any Class Reference

Public Member Functions

- constexpr [any](#) () noexcept
- template<typename _Tp, typename _VTp = _Decay_if_not_any<_Tp>, typename _Mgr = _Manager<_VTp>, [enable_if_t](#)<is_copy_constructible_v<_VTp> &&!__is_in_place_type_v<_VTp>, bool> = true>
[any](#) (_Tp &&__value)
- [any](#) ([any](#) &&__other) noexcept
- [any](#) (const [any](#) &__other)
- template<typename _Tp, typename... _Args, typename _VTp = decay_t<_Tp>, typename _Mgr = _Manager<_VTp>, __any_constructible_t<_VTp, _Args &&...> = false>
[any](#) (in_place_type_t<_Tp>, _Args &&... __args)
- template<typename _Tp, typename _Up, typename... _Args, typename _VTp = decay_t<_Tp>, typename _Mgr = _Manager<_VTp>, __any_constructible_t<_VTp, initializer_list<_Up> &, _Args &&...> = false>
[any](#) (in_place_type_t<_Tp>, initializer_list<_Up> __il, _Args &&... __args)
- ~[any](#) ()
- template<typename _Tp, typename... _Args>
__emplace_t<decay_t<_Tp>, _Args...> [emplace](#) (_Args &&... __args)
- template<typename _Tp, typename _Up, typename... _Args>
__emplace_t<decay_t<_Tp>, initializer_list<_Up> &, _Args &&...> [emplace](#) (initializer_list<_Up> __il, _Args &&... __args)
- bool [has_value](#) () const noexcept
- template<typename _Tp>
[enable_if_t](#)<is_copy_constructible<_Decay_if_not_any<_Tp>>::value, [any](#) &> [operator=](#) (_Tp &&__rhs)
- [any](#) & [operator=](#) ([any](#) &&__rhs) noexcept
- [any](#) & [operator=](#) (const [any](#) &__rhs)
- void [reset](#) () noexcept
- void [swap](#) ([any](#) &__rhs) noexcept
- const [type_info](#) & [type](#) () const noexcept

6.202.1 Detailed Description

A type-safe container of any type.

An `any` object's state is either empty or it stores a contained object of CopyConstructible type.

Since

C++17

6.202.2 Constructor & Destructor Documentation

[any](#)() [1/6]

```
constexpr std::any::any ( ) [inline], [constexpr], [noexcept]
```

Default constructor, creates an empty object.

[any](#)() [2/6]

```
std::any::any (
    const any &__other ) [inline]
```

Copy constructor, copies the state of `__other`.

any() [3/6]

```
std::any::any (
    any && __other ) [inline], [noexcept]
```

Move constructor, transfer the state from __other.

Postcondition

!__other.has_value() (this postcondition is a GNU extension)

any() [4/6]

```
template<typename _Tp , typename _VTP = _Decay_if_not_any<_Tp>, typename _Mgr = _Manager<_VTP>,
enable_if_t< is_copy_constructible_v< _VTP > &&!__is_in_place_type_v< _VTP >, bool > = true>
std::any::any (
    _Tp && __value ) [inline]
```

Construct with a copy of __value as the contained object.

any() [5/6]

```
template<typename _Tp , typename... _Args, typename _VTP = decay_t<_Tp>, typename _Mgr = _←
Manager<_VTP>, __any_constructible_t< _VTP, _Args &&... > = false>
std::any::any (
```

```
    in_place_type_t< _Tp > ,
    _Args &&... __args ) [inline], [explicit]
```

Construct with an object created from __args as the contained object.

any() [6/6]

```
template<typename _Tp , typename _Up , typename... _Args, typename _VTP = decay_t<_Tp>, typename
_Mgr = _Manager<_VTP>, __any_constructible_t< _VTP, initializer_list< _Up > &, _Args &&... > =
false>
```

```
std::any::any (
    in_place_type_t< _Tp > ,
    initializer_list< _Up > __il,
    _Args &&... __args ) [inline], [explicit]
```

Construct with an object created from __il and __args as the contained object.

~any()

```
std::any::~any ( ) [inline]
```

Destructor, calls reset()

6.202.3 Member Function Documentation**emplace()** [1/2]

```
template<typename _Tp , typename... _Args>
__emplace_t< decay_t< _Tp >, _Args... > std::any::emplace (
    _Args &&... __args ) [inline]
```

Emplace with an object created from __args as the contained object.

emplace() [2/2]

```
template<typename _Tp , typename _Up , typename... _Args>
__emplace_t< decay_t< _Tp >, initializer_list< _Up > &, _Args &&... > std::any::emplace (
```

```
initializer_list< _Up > __il,  
_Args &&... __args ) [inline]
```

Emplace with an object created from `__il` and `__args` as the contained object.

has_value()

```
bool std::any::has_value ( ) const [inline], [noexcept]
```

Reports whether there is a contained object or not.

operator=() [1/3]

```
template<typename _Tp >  
enable_if_t< is_copy_constructible< _Decay_if_not_any< _Tp > >::value, any & > std::any::operator=  
(  
    _Tp && __rhs ) [inline]
```

Store a copy of `__rhs` as the contained object.

operator=() [2/3]

```
any & std::any::operator= (  
    any && __rhs ) [inline], [noexcept]
```

Move assignment operator.

Postcondition

`!__rhs.has_value()` (not guaranteed for other implementations)

operator=() [3/3]

```
any & std::any::operator= (  
    const any & __rhs ) [inline]
```

Copy the state of another object.

reset()

```
void std::any::reset ( ) [inline], [noexcept]
```

If not empty, destroy the contained object.

swap()

```
void std::any::swap (  
    any & __rhs ) [inline], [noexcept]
```

Exchange state with another object.

type()

```
const type_info & std::any::type ( ) const [inline], [noexcept]
```

The typeid of the contained object, or typeid(void) if empty.

The documentation for this class was generated from the following file:

- `any`

6.203 std::experimental::fundamentals_v1::any Class Reference

Public Member Functions

- `any ()` noexcept
- `template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>, typename enable_if<
is_constructible<_Tp, _ValueType && >::value, bool >::type = true>
any (_ValueType &&__value)`
- `template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>, typename enable_if<
is_constructible<_Tp, _ValueType && >::value, bool >::type = false>
any (_ValueType &&__value)`
- `any (any &&__other)` noexcept
- `any (const any &__other)`
- `~any ()`
- `void clear ()` noexcept
- `bool empty ()` const noexcept
- `template<typename _ValueType >
enable_if_t<!is_same< any, decay_t<_ValueType > >::value, any & > operator= (_ValueType &&__rhs)`
- `any & operator= (any &&__rhs)` noexcept
- `any & operator= (const any &__rhs)`
- `void swap (any &__rhs)` noexcept
- `const type_info & type ()` const noexcept

Static Public Member Functions

- `template<typename _Tp >
static constexpr bool __is_valid_cast ()`

Friends

- `template<typename _Tp >
enable_if_t< is_object<_Tp >::value, void * > __any_caster (const any *__any)`

6.203.1 Detailed Description

A type-safe container of any type.

An `any` object's state is either empty or it stores a contained object of CopyConstructible type.

6.203.2 Constructor & Destructor Documentation

`any()` [1/5]

```
std::experimental::fundamentals_v1::any::any ( ) [inline], [noexcept]
```

Default constructor, creates an empty object.

`any()` [2/5]

```
std::experimental::fundamentals_v1::any::any (
    const any & __other ) [inline]
```

Copy constructor, copies the state of `__other`.

any() [3/5]

```
std::experimental::fundamentals_v1::any::any (
    any && __other ) [inline], [noexcept]
```

Move constructor, transfer the state from __other.

Postcondition

__other.empty() (this postcondition is a GNU extension)

any() [4/5]

```
template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>,
typename enable_if< is_constructible< _Tp, _ValueType && >::value, bool >::type = true>
std::experimental::fundamentals_v1::any::any (
    _ValueType && __value ) [inline]
```

Construct with a copy of __value as the contained object.

any() [5/5]

```
template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>,
typename enable_if<!is_constructible< _Tp, _ValueType && >::value, bool >::type = false>
std::experimental::fundamentals_v1::any::any (
    _ValueType && __value ) [inline]
```

Construct with a copy of __value as the contained object.

~any()

```
std::experimental::fundamentals_v1::any::~~any ( ) [inline]
```

Destructor, calls clear()

6.203.3 Member Function Documentation**clear()**

```
void std::experimental::fundamentals_v1::any::clear ( ) [inline], [noexcept]
```

If not empty, destroy the contained object.

empty()

```
bool std::experimental::fundamentals_v1::any::empty ( ) const [inline], [noexcept]
```

Reports whether there is a contained object or not.

operator=() [1/3]

```
template<typename _ValueType >
enable_if_t<!is_same< any, decay_t< _ValueType > >::value, any & > std::experimental::fundamentals_v1::any::operator= (
    _ValueType && __rhs ) [inline]
```

Store a copy of __rhs as the contained object.

operator=() [2/3]

```
any & std::experimental::fundamentals_v1::any::operator= (
    any && __rhs ) [inline], [noexcept]
```

Move assignment operator.

Postcondition

`__rhs.empty()` (not guaranteed for other implementations)

operator=() [3/3]

```
any & std::experimental::fundamentals_v1::any::operator= (
    const any & __rhs ) [inline]
```

Copy the state of another object.

swap()

```
void std::experimental::fundamentals_v1::any::swap (
    any & __rhs ) [inline], [noexcept]
```

Exchange state with another object.

type()

```
const type_info & std::experimental::fundamentals_v1::any::type ( ) const [inline], [noexcept]
```

The `typeid` of the contained object, or `typeid(void)` if empty.

The documentation for this class was generated from the following file:

- [experimental/any](#)

6.204 `std::array<_Tp, _Nm>` Struct Template Reference**Public Types**

- `typedef __array_traits<_Tp, _Nm> _AT_Type`
- `typedef const value_type * const_iterator`
- `typedef const value_type * const_pointer`
- `typedef const value_type & const_reference`
- `typedef std::reverse_iterator< const_iterator > const_reverse_iterator`
- `typedef std::ptrdiff_t difference_type`
- `typedef value_type * iterator`
- `typedef value_type * pointer`
- `typedef value_type & reference`
- `typedef std::reverse_iterator< iterator > reverse_iterator`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `constexpr reference at (size_type __n)`
- `constexpr const_reference at (size_type __n) const`
- `constexpr const_reference back () const noexcept`
- `constexpr reference back () noexcept`
- `constexpr const_iterator begin () const noexcept`
- `constexpr iterator begin () noexcept`
- `constexpr const_iterator cbegin () const noexcept`
- `constexpr const_iterator cend () const noexcept`
- `constexpr const_reverse_iterator crbegin () const noexcept`
- `constexpr const_reverse_iterator crend () const noexcept`
- `constexpr const_pointer data () const noexcept`

- constexpr pointer **data** () noexcept
- constexpr bool **empty** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr iterator **end** () noexcept
- constexpr void **fill** (const value_type &__u)
- constexpr const_reference **front** () const noexcept
- constexpr reference **front** () noexcept
- constexpr size_type **max_size** () const noexcept
- constexpr const_reference **operator[]** (size_type __n) const noexcept
- constexpr reference **operator[]** (size_type __n) noexcept
- constexpr [const_reverse_iterator](#) **rbegin** () const noexcept
- constexpr [reverse_iterator](#) **rbegin** () noexcept
- constexpr [const_reverse_iterator](#) **rend** () const noexcept
- constexpr [reverse_iterator](#) **rend** () noexcept
- constexpr size_type **size** () const noexcept
- constexpr void **swap** ([array](#) &__other) noexcept(_AT_Type::_Is_nothrow_swappable::value)

Public Attributes

- `_AT_Type::Type` **_M_elems**

6.204.1 Detailed Description

`template<typename _Tp, std::size_t _Nm>`
`struct std::array<_Tp, _Nm >`

A standard container for storing a fixed size sequence of elements.

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#).
 Sets support random access iterators.

Template Parameters

<i>Tp</i>	Type of element. Required to be a complete type.
<i>Nm</i>	Number of elements.

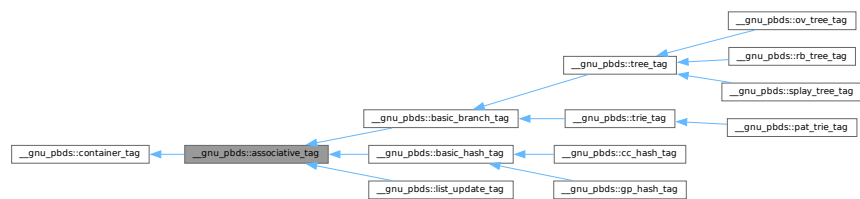
The documentation for this struct was generated from the following file:

- [array](#)

6.205 __gnu_pbds::associative_tag Struct Reference

`#include <tag_and_trait.hpp>`

Inheritance diagram for `__gnu_pbds::associative_tag`:



6.205.1 Detailed Description

Basic associative-container.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.206 std::atomic<_Tp> Class Template Reference

Public Types

- using **value_type** = _Tp

Public Member Functions

- constexpr **atomic** (_Tp __i) noexcept
- **atomic** (const [atomic](#) &)=delete
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) volatile noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) volatile noexcept
- _Tp **exchange** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- _Tp **exchange** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- _Tp **load** ([memory_order](#) __m=memory_order_seq_cst) const noexcept
- _Tp **load** ([memory_order](#) __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator _Tp** () const noexcept
- **operator _Tp** () const volatile noexcept
- _Tp **operator=** (_Tp __i) noexcept
- _Tp **operator=** (_Tp __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- void **store** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- void **store** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- void **wait** (_Tp __old, [memory_order](#) __m=memory_order_seq_cst) const noexcept

Static Public Attributes

- static constexpr bool **is_always_lock_free**

6.206.1 Detailed Description

```
template<typename _Tp>
class std::atomic<_Tp>
```

Generic atomic type, primary class template.

Template Parameters

<code>_Tp</code>	Type to be made atomic, must be trivially copyable.
------------------	---

The documentation for this class was generated from the following file:

- [atomic](#)

6.207 `std::atomic<_Tp * >` Struct Template Reference

Public Types

- typedef `__atomic_base<_Tp * >` `__base_type`
- typedef `_Tp *` `__pointer_type`
- using `difference_type` = `ptrdiff_t`
- using `value_type` = `_Tp *`

Public Member Functions

- constexpr `atomic` (`__pointer_type` `__p`) noexcept
- `atomic` (const `atomic` &)=delete
- bool `compare_exchange_strong` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m1`, `memory_order` `__m2`) noexcept
- bool `compare_exchange_strong` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m1`, `memory_order` `__m2`) volatile noexcept
- bool `compare_exchange_strong` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m`=`memory_order_seq_cst`) noexcept
- bool `compare_exchange_strong` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m`=`memory_order_seq_cst`) volatile noexcept
- bool `compare_exchange_weak` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m1`, `memory_order` `__m2`) noexcept
- bool `compare_exchange_weak` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m1`, `memory_order` `__m2`) volatile noexcept
- bool `compare_exchange_weak` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m`=`memory_order_seq_cst`) noexcept
- bool `compare_exchange_weak` (`__pointer_type` &`__p1`, `__pointer_type` `__p2`, `memory_order` `__m`=`memory_order_seq_cst`) volatile noexcept
- `__pointer_type` `exchange` (`__pointer_type` `__p`, `memory_order` `__m`=`memory_order_seq_cst`) noexcept
- `__pointer_type` `exchange` (`__pointer_type` `__p`, `memory_order` `__m`=`memory_order_seq_cst`) volatile noexcept
- `__pointer_type` `fetch_add` (`ptrdiff_t` `__d`, `memory_order` `__m`=`memory_order_seq_cst`) noexcept
- `__pointer_type` `fetch_add` (`ptrdiff_t` `__d`, `memory_order` `__m`=`memory_order_seq_cst`) volatile noexcept
- `__pointer_type` `fetch_sub` (`ptrdiff_t` `__d`, `memory_order` `__m`=`memory_order_seq_cst`) noexcept
- `__pointer_type` `fetch_sub` (`ptrdiff_t` `__d`, `memory_order` `__m`=`memory_order_seq_cst`) volatile noexcept
- bool `is_lock_free` () const noexcept
- bool `is_lock_free` () const volatile noexcept
- `__pointer_type` `load` (`memory_order` `__m`=`memory_order_seq_cst`) const noexcept
- `__pointer_type` `load` (`memory_order` `__m`=`memory_order_seq_cst`) const volatile noexcept
- void `notify_all` () noexcept
- void `notify_one` () noexcept
- `operator __pointer_type` () const noexcept
- `operator __pointer_type` () const volatile noexcept
- `__pointer_type` `operator++` () noexcept
- `__pointer_type` `operator++` () volatile noexcept

- `__pointer_type operator++` (int) noexcept
- `__pointer_type operator++` (int) volatile noexcept
- `__pointer_type operator+=` (ptrdiff_t __d) noexcept
- `__pointer_type operator+=` (ptrdiff_t __d) volatile noexcept
- `__pointer_type operator--` () noexcept
- `__pointer_type operator--` () volatile noexcept
- `__pointer_type operator--` (int) noexcept
- `__pointer_type operator--` (int) volatile noexcept
- `__pointer_type operator-=` (ptrdiff_t __d) noexcept
- `__pointer_type operator-=` (ptrdiff_t __d) volatile noexcept
- `__pointer_type operator=` (__pointer_type __p) noexcept
- `__pointer_type operator=` (__pointer_type __p) volatile noexcept
- `atomic & operator=` (const `atomic` &) volatile=delete
- `atomic & operator=` (const `atomic` &)=delete
- void `store` (__pointer_type __p, `memory_order` __m=memory_order_seq_cst) noexcept
- void `store` (__pointer_type __p, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- void `wait` (__pointer_type __old, `memory_order` __m=memory_order_seq_cst) const noexcept

Public Attributes

- `__base_type _M_b`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.207.1 Detailed Description

```
template<typename _Tp>
struct std::atomic< _Tp * >
```

Partial specialization for pointer types.

The documentation for this struct was generated from the following file:

- `atomic`

6.208 `std::atomic< bool >` Struct Reference

Public Types

- using `value_type` = bool

Public Member Functions

- constexpr `atomic` (bool __i) noexcept
- `atomic` (const `atomic` &)=delete
- bool `compare_exchange_strong` (bool &__i1, bool __i2, `memory_order` __m1, `memory_order` __m2) noexcept
- bool `compare_exchange_strong` (bool &__i1, bool __i2, `memory_order` __m1, `memory_order` __m2) volatile noexcept
- bool `compare_exchange_strong` (bool &__i1, bool __i2, `memory_order` __m=memory_order_seq_cst) noexcept
- bool `compare_exchange_strong` (bool &__i1, bool __i2, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- bool `compare_exchange_weak` (bool &__i1, bool __i2, `memory_order` __m1, `memory_order` __m2) noexcept

- `bool compare_exchange_weak (bool &__i1, bool __i2, memory_order __m1, memory_order __m2) volatile noexcept`
- `bool compare_exchange_weak (bool &__i1, bool __i2, memory_order __m=memory_order_seq_cst) noexcept`
- `bool compare_exchange_weak (bool &__i1, bool __i2, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `bool exchange (bool __i, memory_order __m=memory_order_seq_cst) noexcept`
- `bool exchange (bool __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `bool is_lock_free () const noexcept`
- `bool is_lock_free () const volatile noexcept`
- `bool load (memory_order __m=memory_order_seq_cst) const noexcept`
- `bool load (memory_order __m=memory_order_seq_cst) const volatile noexcept`
- `void notify_all () noexcept`
- `void notify_one () noexcept`
- `operator bool () const noexcept`
- `operator bool () const volatile noexcept`
- `bool operator= (bool __i) noexcept`
- `bool operator= (bool __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `void store (bool __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (bool __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (bool __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- `static constexpr bool is_always_lock_free`

6.208.1 Detailed Description

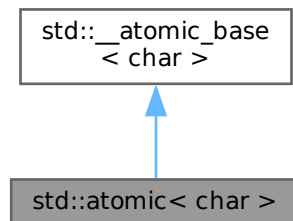
`atomic<bool>`

The documentation for this struct was generated from the following file:

- [atomic](#)

6.209 std::atomic< char > Struct Reference

Inheritance diagram for `std::atomic< char >`:



Public Types

- typedef [__atomic_base](#)< char > [__base_type](#)
- typedef char [__integral_type](#)
- using [difference_type](#) = value_type
- using [value_type](#) = char

Public Member Functions

- constexpr **atomic** ([__integral_type](#) __i) noexcept
- **atomic** (const [atomic](#) &)=delete
- bool **compare_exchange_strong** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m1, [memory_order](#) __m2) noexcept
- bool **compare_exchange_strong** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m1, [memory_order](#) __m2) volatile noexcept
- bool **compare_exchange_strong** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m1, [memory_order](#) __m2) noexcept
- bool **compare_exchange_weak** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m1, [memory_order](#) __m2) volatile noexcept
- bool **compare_exchange_weak** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** ([__int_type](#) &__i1, [__int_type](#) __i2, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **exchange** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **exchange** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **fetch_add** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **fetch_add** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **fetch_and** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **fetch_and** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **fetch_or** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **fetch_or** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **fetch_sub** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **fetch_sub** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- [__int_type](#) **fetch_xor** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- [__int_type](#) **fetch_xor** ([__int_type](#) __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- [__int_type](#) **load** ([memory_order](#) __m=memory_order_seq_cst) const noexcept
- [__int_type](#) **load** ([memory_order](#) __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** [__int_type](#) () const noexcept
- **operator** [__int_type](#) () const volatile noexcept
- [__int_type](#) **operator&=** ([__int_type](#) __i) noexcept
- [__int_type](#) **operator&=** ([__int_type](#) __i) volatile noexcept
- [__int_type](#) **operator++** () noexcept
- [__int_type](#) **operator++** () volatile noexcept
- [__int_type](#) **operator++** (int) noexcept

- `__int_type operator++ (int) volatile noexcept`
- `__int_type operator+= (__int_type __i) noexcept`
- `__int_type operator+= (__int_type __i) volatile noexcept`
- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator-- (int) noexcept`
- `__int_type operator-- (int) volatile noexcept`
- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.209.1 Detailed Description

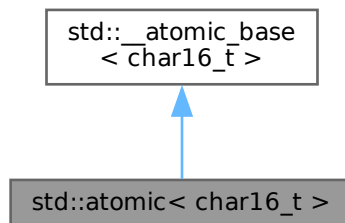
Explicit specialization for char.

The documentation for this struct was generated from the following file:

- `atomic`

6.210 std::atomic< char16_t > Struct Reference

Inheritance diagram for `std::atomic< char16_t >`:



Public Types

- typedef `__atomic_base< char16_t > __base_type`
- typedef `char16_t __integral_type`
- using `difference_type` = `value_type`
- using `value_type` = `char16_t`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.210.1 Detailed Description

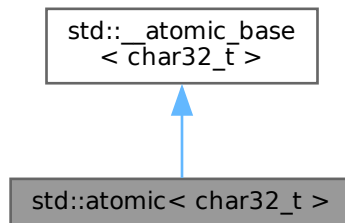
Explicit specialization for `char16_t`.

The documentation for this struct was generated from the following file:

- `atomic`

6.211 `std::atomic< char32_t >` Struct Reference

Inheritance diagram for `std::atomic< char32_t >`:



Public Types

- typedef `__atomic_base< char32_t > __base_type`
- typedef `char32_t __integral_type`
- using `difference_type = value_type`
- using `value_type = char32_t`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.211.1 Detailed Description

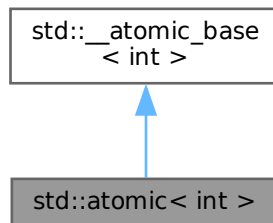
Explicit specialization for `char32_t`.

The documentation for this struct was generated from the following file:

- `atomic`

6.212 `std::atomic< int >` Struct Reference

Inheritance diagram for `std::atomic< int >`:



Public Types

- typedef `__atomic_base< int > __base_type`
- typedef int `__integral_type`
- using `difference_type` = `value_type`
- using `value_type` = int

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.212.1 Detailed Description

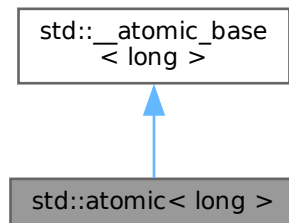
Explicit specialization for int.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.213 std::atomic< long > Struct Reference

Inheritance diagram for std::atomic< long >:



Public Types

- typedef `__atomic_base< long > __base_type`
- typedef long `__integral_type`
- using `difference_type` = value_type
- using `value_type` = long

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.213.1 Detailed Description

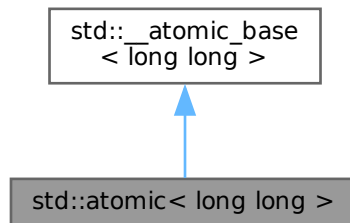
Explicit specialization for long.

The documentation for this struct was generated from the following file:

- `atomic`

6.214 std::atomic< long long > Struct Reference

Inheritance diagram for `std::atomic< long long >`:



Public Types

- typedef `__atomic_base< long long >` `__base_type`
- typedef `long long` `__integral_type`
- using `difference_type` = `value_type`
- using `value_type` = `long long`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- `static constexpr bool is_always_lock_free`

6.214.1 Detailed Description

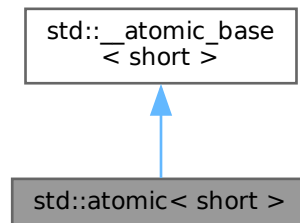
Explicit specialization for long long.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.215 std::atomic< short > Struct Reference

Inheritance diagram for `std::atomic< short >`:



Public Types

- `typedef __atomic_base< short > __base_type`
- `typedef short __integral_type`
- `using difference_type = value_type`
- `using value_type = short`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.215.1 Detailed Description

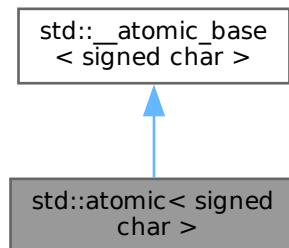
Explicit specialization for short.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.216 std::atomic< signed char > Struct Reference

Inheritance diagram for `std::atomic< signed char >`:



Public Types

- typedef `__atomic_base< signed char > __base_type`
- typedef `signed char __integral_type`
- using `difference_type = value_type`
- using `value_type = signed char`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.216.1 Detailed Description

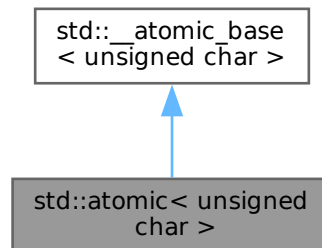
Explicit specialization for signed char.

The documentation for this struct was generated from the following file:

- `atomic`

6.217 `std::atomic< unsigned char >` Struct Reference

Inheritance diagram for `std::atomic< unsigned char >`:



Public Types

- typedef `__atomic_base< unsigned char > __base_type`
- typedef `unsigned char __integral_type`
- using `difference_type = value_type`
- using `value_type = unsigned char`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.217.1 Detailed Description

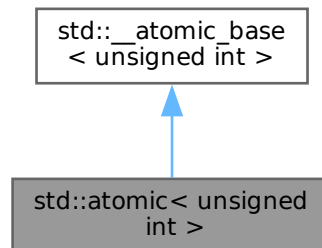
Explicit specialization for unsigned char.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.218 `std::atomic< unsigned int >` Struct Reference

Inheritance diagram for `std::atomic< unsigned int >`:



Public Types

- typedef `__atomic_base< unsigned int > __base_type`
- typedef `unsigned int __integral_type`
- using `difference_type = value_type`
- using `value_type = unsigned int`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.218.1 Detailed Description

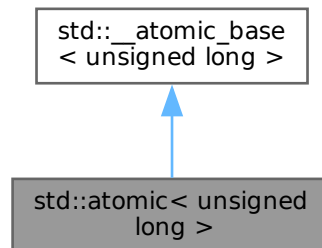
Explicit specialization for unsigned int.

The documentation for this struct was generated from the following file:

- `atomic`

6.219 `std::atomic< unsigned long >` Struct Reference

Inheritance diagram for `std::atomic< unsigned long >`:



Public Types

- typedef `__atomic_base< unsigned long > __base_type`
- typedef `unsigned long __integral_type`
- using `difference_type = value_type`
- using `value_type = unsigned long`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.219.1 Detailed Description

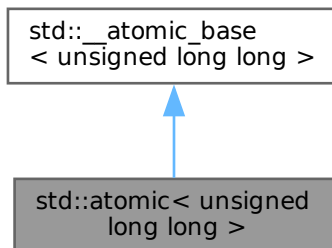
Explicit specialization for unsigned long.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.220 `std::atomic< unsigned long long >` Struct Reference

Inheritance diagram for `std::atomic< unsigned long long >`:



Public Types

- typedef `__atomic_base< unsigned long long > __base_type`
- typedef `unsigned long long __integral_type`
- using `difference_type = value_type`
- using `value_type = unsigned long long`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- static constexpr bool `is_always_lock_free`

6.220.1 Detailed Description

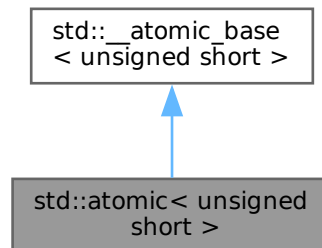
Explicit specialization for unsigned long long.

The documentation for this struct was generated from the following file:

- `atomic`

6.221 `std::atomic< unsigned short >` Struct Reference

Inheritance diagram for `std::atomic< unsigned short >`:



Public Types

- typedef `__atomic_base< unsigned short > __base_type`
- typedef unsigned short `__integral_type`
- using `difference_type` = `value_type`
- using `value_type` = unsigned short

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- `static constexpr bool is_always_lock_free`

6.221.1 Detailed Description

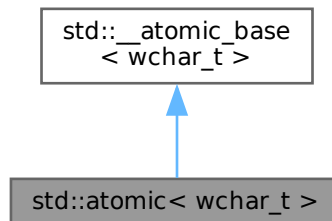
Explicit specialization for unsigned short.

The documentation for this struct was generated from the following file:

- [atomic](#)

6.222 std::atomic< wchar_t > Struct Reference

Inheritance diagram for `std::atomic< wchar_t >`:



Public Types

- `typedef __atomic_base< wchar_t > __base_type`
- `typedef wchar_t __integral_type`
- `using difference_type = value_type`
- `using value_type = wchar_t`

Public Member Functions

- constexpr **atomic** (__integral_type __i) noexcept
- **atomic** (const **atomic** &)=delete
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m1, **memory_order** __m2) volatile noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (__int_type &__i1, __int_type __i2, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **exchange** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_add** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_and** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_or** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_sub** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) noexcept
- __int_type **fetch_xor** (__int_type __i, **memory_order** __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const noexcept
- __int_type **load** (**memory_order** __m=memory_order_seq_cst) const volatile noexcept
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept

- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &) volatile=delete`
- `atomic & operator= (const atomic &)=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (__int_type __i, memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void wait (__int_type __old, memory_order __m=memory_order_seq_cst) const noexcept`

Static Public Attributes

- `static constexpr bool is_always_lock_free`

6.222.1 Detailed Description

Explicit specialization for `wchar_t`.

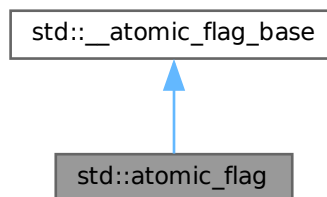
The documentation for this struct was generated from the following file:

- `atomic`

6.223 std::atomic_flag Struct Reference

```
#include <atomic_base.h>
```

Inheritance diagram for `std::atomic_flag`:



Public Member Functions

- `constexpr atomic_flag (bool __i) noexcept`
- `atomic_flag (const atomic_flag &)=delete`
- `void clear (memory_order __m=memory_order_seq_cst) noexcept`
- `void clear (memory_order __m=memory_order_seq_cst) volatile noexcept`
- `void notify_all () noexcept`
- `void notify_one () noexcept`
- `atomic_flag & operator= (const atomic_flag &) volatile=delete`
- `atomic_flag & operator= (const atomic_flag &)=delete`
- `bool test (memory_order __m=memory_order_seq_cst) const noexcept`

- bool **test** ([memory_order](#) __m=memory_order_seq_cst) const volatile noexcept
- bool **test_and_set** ([memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **test_and_set** ([memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- void **wait** (bool __old, [memory_order](#) __m=memory_order_seq_cst) const noexcept

Public Attributes

- [__atomic_flag_data_type](#) **_M_i**

6.223.1 Detailed Description

[atomic_flag](#)

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

6.224 std::atomic_ref< _Tp > Struct Template Reference

Inherits [std::__atomic_ref< _Tp, bool, bool >](#).

Public Member Functions

- **atomic_ref** ([_Tp](#) &__t) noexcept
- **atomic_ref** (const [atomic_ref](#) &)=default
- **atomic_ref** & **operator=** (const [atomic_ref](#) &)=delete

6.224.1 Detailed Description

```
template<typename _Tp>
struct std::atomic_ref< _Tp >
```

Class template to provide atomic operations on a non-atomic variable.
The documentation for this struct was generated from the following file:

- [atomic](#)

6.225 std::auto_ptr< _Tp > Class Template Reference

```
#include <auto_ptr.h>
```

Public Types

- typedef [_Tp](#) [element_type](#)

Public Member Functions

- [auto_ptr](#) ([auto_ptr](#) &__a) throw ()
- template<typename _Tp1 >
 [auto_ptr](#) ([auto_ptr](#)< _Tp1 > &__a) throw ()
- [auto_ptr](#) ([auto_ptr_ref](#)< [element_type](#) > __ref) throw ()
- [auto_ptr](#) ([element_type](#) *__p=0) throw ()
- [~auto_ptr](#) ()
- [element_type](#) * [get](#) () const throw ()
- template<typename _Tp1 >
 operator auto_ptr () throw ()

- `template<typename _Tp1 >`
`operator auto_ptr_ref () throw ()`
- `element_type & operator* () const throw ()`
- `element_type * operator-> () const throw ()`
- `auto_ptr & operator= (auto_ptr &__a) throw ()`
- `template<typename _Tp1 >`
`auto_ptr & operator= (auto_ptr< _Tp1 > &__a) throw ()`
- `auto_ptr & operator= (auto_ptr_ref< element_type > __ref) throw ()`
- `element_type * release () throw ()`
- `void reset (element_type * __p=0) throw ()`

6.225.1 Detailed Description

`template<typename _Tp>`
`class std::auto_ptr< _Tp >`

A simple smart pointer providing strict ownership semantics.
The Standard says:

An `auto_ptr` owns the object it holds a pointer to. Copying an `auto_ptr` copies the pointer and transfers ownership to the destination. If more than one `auto_ptr` owns the same object at the same time the behavior of the program is undefined.

The uses of `auto_ptr` include providing temporary exception-safety for dynamically allocated memory, passing ownership of dynamically allocated memory to a function, and returning dynamically allocated memory from a function. `auto_ptr` does not meet the CopyConstructible requirements for Standard Library `container` elements and thus instantiating a Standard Library container with an `auto_ptr` results in undefined behavior.

Quoted from [20.4.5]/3.

Good examples of what can and cannot be done with `auto_ptr` can be found in the libstdc++ testsuite.

`_GLIBCXX_RESOLVE_LIB_DEFECTS`

1. `auto_ptr<>` conversion issues These resolutions have all been incorporated.

6.225.2 Member Typedef Documentation

`element_type`

`template<typename _Tp >`
`typedef _Tp std::auto_ptr< _Tp >::element_type`
The pointed-to type.

6.225.3 Constructor & Destructor Documentation

`auto_ptr()` [1/4]

`template<typename _Tp >`
`std::auto_ptr< _Tp >::auto_ptr (`
`element_type * __p = 0) throw ()` [inline], [explicit]

An `auto_ptr` is usually constructed from a raw pointer.

Parameters

<code>__p</code>	A pointer (defaults to NULL).
------------------	-------------------------------

This object now *owns* the object pointed to by `__p`.

auto_ptr() [2/4]

```
template<typename _Tp >
std::auto_ptr<_Tp >::auto_ptr (
    auto_ptr<_Tp > & __a ) throw ( )    [inline]
```

An auto_ptr can be constructed from another auto_ptr.

Parameters

<code>__a</code>	Another auto_ptr of the same type.
------------------	------------------------------------

This object now *owns* the object previously owned by `__a`, which has given up ownership.

auto_ptr() [3/4]

```
template<typename _Tp >
template<typename _Tp1 >
std::auto_ptr<_Tp >::auto_ptr (
    auto_ptr<_Tp1 > & __a ) throw ( )    [inline]
```

An auto_ptr can be constructed from another auto_ptr.

Parameters

<code>__a</code>	Another auto_ptr of a different but related type.
------------------	---

A pointer-to-Tp1 must be convertible to a pointer-to-Tp/element_type.

This object now *owns* the object previously owned by `__a`, which has given up ownership.

~auto_ptr()

```
template<typename _Tp >
std::auto_ptr<_Tp >::~~auto_ptr ( )    [inline]
```

When the auto_ptr goes out of scope, the object it owns is deleted. If it no longer owns anything (i.e., `get()` is NULL), then this has no effect.

The C++ standard says there is supposed to be an empty throw specification here, but omitting it is standard conforming. Its presence can be detected only if `_Tp::~~_Tp()` throws, but this is prohibited. [17.4.3.6]/2

auto_ptr() [4/4]

```
template<typename _Tp >
std::auto_ptr<_Tp >::auto_ptr (
    auto_ptr_ref<element_type > __ref ) throw ( )    [inline]
```

Automatic conversions.

These operations are supposed to convert an auto_ptr into and from an auto_ptr_ref automatically as needed. This would allow constructs such as

```
auto_ptr<Derived> func_returning_auto_ptr(.....);
...
auto_ptr<Base> ptr = func_returning_auto_ptr(.....);
```

But it doesn't work, and won't be fixed. For further details see <http://cplusplus.github.io/LWG/lwg-closed.html#463>

6.225.4 Member Function Documentation

get()

```
template<typename _Tp >
element_type * std::auto_ptr< _Tp >::get ( ) const throw ( )    [inline]
```

Bypassing the smart pointer.

Returns

The raw pointer being managed.

You can get a copy of the pointer that this object owns, for situations such as passing to a function which only accepts a raw pointer.

Note

This `auto_ptr` still owns the memory.

operator*()

```
template<typename _Tp >
element_type & std::auto_ptr< _Tp >::operator* ( ) const throw ( )    [inline]
```

Smart pointer dereferencing.

If this `auto_ptr` no longer owns anything, then this operation will crash. (For a smart pointer, *no longer owns anything* is the same as being a null pointer, and you know what happens when you dereference one of those...)

operator->()

```
template<typename _Tp >
element_type * std::auto_ptr< _Tp >::operator-> ( ) const throw ( )    [inline]
```

Smart pointer dereferencing.

This returns the pointer itself, which the language then will automatically cause to be dereferenced.

operator=() [1/2]

```
template<typename _Tp >
auto_ptr & std::auto_ptr< _Tp >::operator= (
    auto_ptr< _Tp > & __a ) throw ( )    [inline]
```

`auto_ptr` assignment operator.

Parameters

<code>__a</code>	Another <code>auto_ptr</code> of the same type.
------------------	---

This object now *owns* the object previously owned by `__a`, which has given up ownership. The object that this one *used* to own and track has been deleted.

References `std::auto_ptr< _Tp >::reset()`.

operator=() [2/2]

```
template<typename _Tp >
template<typename _Tp1 >
auto_ptr & std::auto_ptr< _Tp >::operator= (
    auto_ptr< _Tp1 > & __a ) throw ( )    [inline]
```

`auto_ptr` assignment operator.

Parameters

<code>__a</code>	Another <code>auto_ptr</code> of a different but related type.
------------------	--

A pointer-to-`Tp1` must be convertible to a pointer-to-`Tp/element_type`.

This object now *owns* the object previously owned by `__a`, which has given up ownership. The object that this one *used* to own and track has been deleted.

References `std::auto_ptr<_Tp>::reset()`.

release()

```
template<typename _Tp >
element_type * std::auto_ptr<_Tp>::release ( ) throw ( )    [inline]
```

Bypassing the smart pointer.

Returns

The raw pointer being managed.

You can get a copy of the pointer that this object owns, for situations such as passing to a function which only accepts a raw pointer.

Note

This `auto_ptr` no longer owns the memory. When this object goes out of scope, nothing will happen.

reset()

```
template<typename _Tp >
void std::auto_ptr<_Tp>::reset (
    element_type * __p = 0 ) throw ( )    [inline]
```

Forcibly deletes the managed object.

Parameters

<code>__p</code>	A pointer (defaults to NULL).
------------------	-------------------------------

This object now *owns* the object pointed to by `__p`. The previous object has been deleted.

Referenced by `std::auto_ptr<_Tp>::operator=()`, and `std::auto_ptr<_Tp>::operator=()`.

The documentation for this class was generated from the following files:

- [shared_ptr_base.h](#)
- [auto_ptr.h](#)

6.226 `std::auto_ptr_ref<_Tp1>` > Struct Template Reference

```
#include <auto_ptr.h>
```

Public Member Functions

- `auto_ptr_ref (_Tp1 * __p)`

Public Attributes

- `_Tp1 * _M_ptr`

6.226.1 Detailed Description

```
template<typename _Tp1>
struct std::auto_ptr_ref<_Tp1 >
```

A wrapper class to provide `auto_ptr` with reference semantics. For example, an `auto_ptr` can be assigned (or constructed from) the result of a function which returns an `auto_ptr` by value.

All the `auto_ptr_ref` stuff should happen behind the scenes.

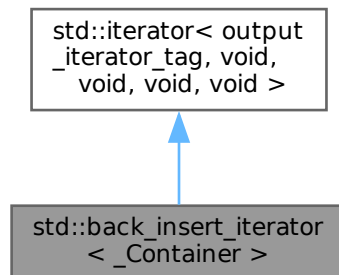
The documentation for this struct was generated from the following file:

- [auto_ptr.h](#)

6.227 `std::back_insert_iterator<_Container>` Class Template Reference

```
#include <stl_iterator.h>
```

Inheritance diagram for `std::back_insert_iterator<_Container>`:



Public Types

- typedef `_Container` [container_type](#)
- using **difference_type** = `ptrdiff_t`
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

Public Member Functions

- constexpr [back_insert_iterator](#) (`_Container &__x`)
- constexpr [back_insert_iterator](#) & [operator*](#) ()
- constexpr [back_insert_iterator](#) & [operator++](#) ()
- constexpr [back_insert_iterator](#) [operator++](#) (int)
- constexpr [back_insert_iterator](#) & [operator=](#) (const typename `_Container::value_type` &__value)
- constexpr [back_insert_iterator](#) & **[operator=](#)** (typename `_Container::value_type` &&__value)

Protected Attributes

- `_Container * container`

6.227.1 Detailed Description

```
template<typename _Container>
class std::back_insert_iterator<_Container>
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator appends it to the container using `push_back`.

Tip: Using the `back_inserter` function to create these iterators can save typing.

6.227.2 Member Typedef Documentation**container_type**

```
template<typename _Container>
typedef _Container std::back_insert_iterator<_Container>::container_type
```

A nested typedef for the type of whatever container you used.

iterator_category

```
typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >←
::iterator_category [inherited]
```

One of the [tag types](#).

pointer

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer [inherited]
```

This type represents a pointer-to-value_type.

reference

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference [inherited]
```

This type represents a reference-to-value_type.

value_type

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type [inherited]
```

The type "pointed to" by the iterator.

6.227.3 Constructor & Destructor Documentation**back_insert_iterator()**

```
template<typename _Container>
constexpr std::back_insert_iterator<_Container>::back_insert_iterator (
    _Container & __x ) [inline], [explicit], [constexpr]
```

The only way to create this iterator is with a container.

6.227.4 Member Function Documentation**operator*()**

```
template<typename _Container>
```

```
constexpr back_insert_iterator & std::back_insert_iterator< _Container >::operator* ( ) [inline],
[constexpr]
```

Simply returns `*this`.

operator++() [1/2]

```
template<typename _Container >
constexpr back_insert_iterator & std::back_insert_iterator< _Container >::operator++ ( ) [inline],
[constexpr]
```

Simply returns `*this`. (This iterator does not *move*.)

operator++() [2/2]

```
template<typename _Container >
constexpr back_insert_iterator std::back_insert_iterator< _Container >::operator++ (
    int ) [inline], [constexpr]
```

Simply returns `*this`. (This iterator does not *move*.)

operator=()

```
template<typename _Container >
constexpr back_insert_iterator & std::back_insert_iterator< _Container >::operator= (
    const typename _Container::value_type & __value ) [inline], [constexpr]
```

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const <code>T</code> for <code>container<T></code> .
----------------------	--

Returns

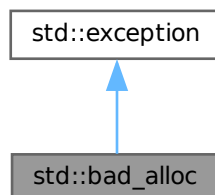
This iterator, for chained operations.

This kind of iterator doesn't really have a *position* in the container (you can think of the position as being permanently at the end, if you like). Assigning a value to the iterator will always append the value to the end of the container. The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.228 std::bad_alloc Class Reference

Inheritance diagram for `std::bad_alloc`:



Public Member Functions

- `bad_alloc` (const `bad_alloc` &)=default
- `bad_alloc` & `operator=` (const `bad_alloc` &)=default
- virtual const char * `what` () const throw ()

6.228.1 Detailed Description

Exception possibly thrown by `new`.

`bad_alloc` (or classes derived from it) is used to report allocation errors from the throwing forms of `new`.

6.228.2 Member Function Documentation

`what()`

```
virtual const char * std::bad_alloc::what ( ) const throw ( ) [virtual]
```

Returns a C-style character string describing the general cause of the current error.

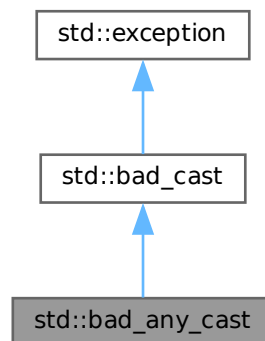
Reimplemented from `std::exception`.

The documentation for this class was generated from the following file:

- `new`

6.229 `std::bad_any_cast` Class Reference

Inheritance diagram for `std::bad_any_cast`:



Public Member Functions

- virtual const char * `what` () const noexcept

6.229.1 Detailed Description

Exception class thrown by a failed `any_cast`.

6.229.2 Member Function Documentation

what()

```
virtual const char * std::bad_any_cast::what ( ) const [inline], [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

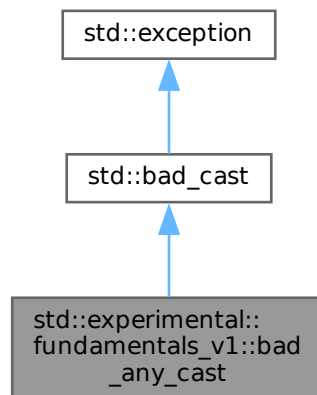
Reimplemented from [std::bad_cast](#).

The documentation for this class was generated from the following file:

- [any](#)

6.230 std::experimental::fundamentals_v1::bad_any_cast Class Reference

Inheritance diagram for `std::experimental::fundamentals_v1::bad_any_cast`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.230.1 Detailed Description

Exception class thrown by a failed `any_cast`.

6.230.2 Member Function Documentation

what()

```
virtual const char * std::experimental::fundamentals_v1::bad_any_cast::what ( ) const [inline], [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

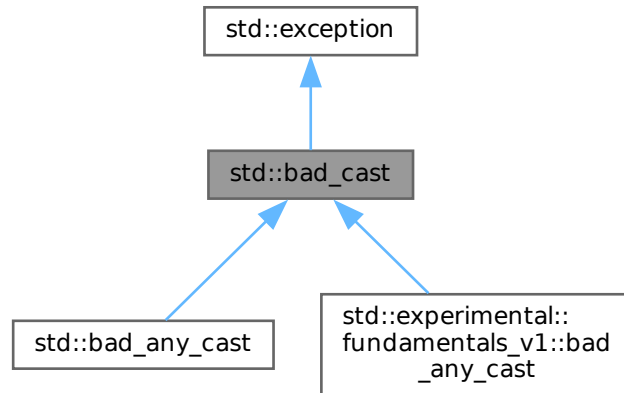
Reimplemented from [std::bad_cast](#).

The documentation for this class was generated from the following file:

- [experimental/any](#)

6.231 `std::bad_cast` Class Reference

Inheritance diagram for `std::bad_cast`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.231.1 Detailed Description

Thrown during incorrect typecasting.

If you attempt an invalid `dynamic_cast` expression, an instance of this class (or something derived from this class) is thrown.

6.231.2 Member Function Documentation

what()

```
virtual const char * std::bad_cast::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

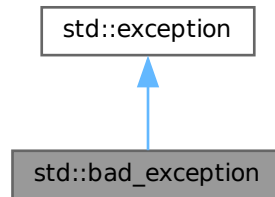
Reimplemented in [std::bad_any_cast](#), and [std::experimental::fundamentals_v1::bad_any_cast](#).

The documentation for this class was generated from the following file:

- [typeinfo](#)

6.232 std::bad_exception Class Reference

Inheritance diagram for std::bad_exception:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.232.1 Detailed Description

If an exception is thrown which is not listed in a function's exception specification, one of these may be thrown.

6.232.2 Member Function Documentation

what()

```
virtual const char * std::bad_exception::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

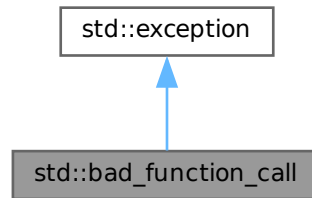
The documentation for this class was generated from the following file:

- [exception](#)

6.233 std::bad_function_call Class Reference

```
#include <std_function.h>
```

Inheritance diagram for std::bad_function_call:



Public Member Functions

- const char * [what](#) () const noexcept

6.233.1 Detailed Description

Exception class thrown when class template function's operator() is called with an empty target.

6.233.2 Member Function Documentation

what()

```
const char * std::bad_function_call::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

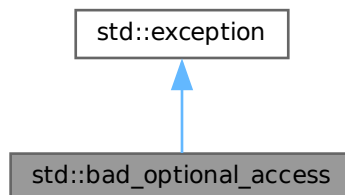
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [std_function.h](#)

6.234 std::bad_optional_access Class Reference

Inheritance diagram for std::bad_optional_access:



Public Member Functions

- `const char * what () const` noexcept override

6.234.1 Detailed Description

Exception class thrown when a disengaged optional object is dereferenced.

6.234.2 Member Function Documentation

`what()`

`const char * std::bad_optional_access::what () const` `[inline]`, `[override]`, `[virtual]`, `[noexcept]`
Returns a C-style character string describing the general cause of the current error.

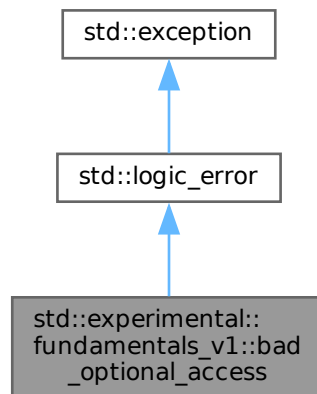
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [optional](#)

6.235 std::experimental::fundamentals_v1::bad_optional_access Class Reference

Inheritance diagram for `std::experimental::fundamentals_v1::bad_optional_access`:



Public Member Functions

- `bad_optional_access (const char * __arg)`
- `virtual const char * what () const` noexcept

6.235.1 Detailed Description

Exception class thrown when a disengaged optional object is dereferenced.

6.235.2 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

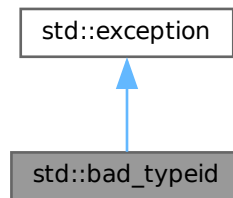
Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [experimental/optional](#)

6.236 std::bad_typeid Class Reference

Inheritance diagram for std::bad_typeid:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.236.1 Detailed Description

Thrown when a NULL pointer in a `typeid` expression is used.

6.236.2 Member Function Documentation

what()

```
virtual const char * std::bad_typeid::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

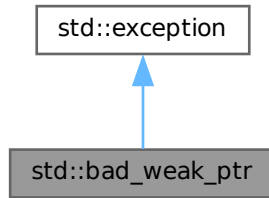
The documentation for this class was generated from the following file:

- [typeinfo](#)

6.237 std::bad_weak_ptr Class Reference

```
#include <shared_ptr_base.h>
```

Inheritance diagram for `std::bad_weak_ptr`:



Public Member Functions

- virtual `char const * what () const` noexcept

6.237.1 Detailed Description

Exception possibly thrown by `shared_ptr`.

6.237.2 Member Function Documentation

`what()`

```
virtual char const * std::bad_weak_ptr::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

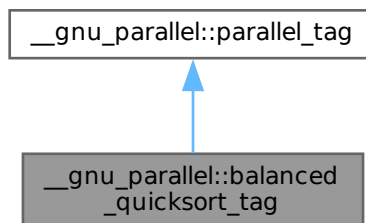
The documentation for this class was generated from the following file:

- [shared_ptr_base.h](#)

6.238 `__gnu_parallel::balanced_quicksort_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::balanced_quicksort_tag`:



Public Member Functions

- **balanced_quicksort_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **__get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

6.238.1 Detailed Description

Forces parallel sorting using balanced quicksort at compile time.

6.238.2 Member Function Documentation**__get_num_threads()**

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#).

set_num_threads()

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

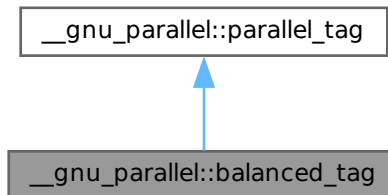
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.239 `__gnu_parallel::balanced_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::balanced_tag`:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

6.239.1 Detailed Description

Recommends parallel execution using dynamic load-balancing at compile time.

6.239.2 Member Function Documentation

[__get_num_threads\(\)](#)

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#)

[set_num_threads\(\)](#)

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.240 std::tr2::bases<_Tp> Struct Template Reference

Public Types

- typedef [__reflection_typelist](#)< __bases(_Tp)... > **type**

6.240.1 Detailed Description

```
template<typename _Tp>
struct std::tr2::bases<_Tp>
```

Sequence abstraction metafunctions for manipulating a typelist.

Enumerate all the base classes of a class. Form of a typelist.

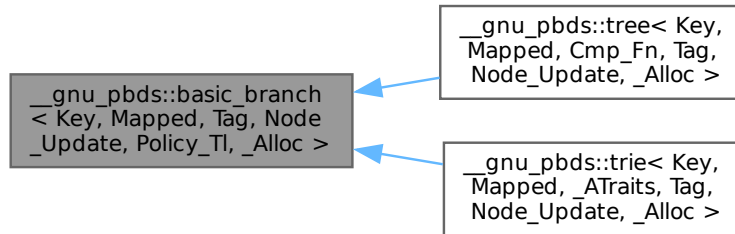
The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

6.241 __gnu_pbds::basic_branch<Key, Mapped, Tag, Node_Update, Policy_TI, _Alloc> Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >`:



Public Types

- typedef Node_Update **node_update**

Protected Member Functions

- **basic_branch** (const [basic_branch](#) &other)
- template<typename T0 >
basic_branch (T0 t0)
- template<typename T0 , typename T1 >
basic_branch (T0 t0, T1 t1)
- template<typename T0 , typename T1 , typename T2 >
basic_branch (T0 t0, T1 t1, T2 t2)
- template<typename T0 , typename T1 , typename T2 , typename T3 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6)

6.241.1 Detailed Description

template<typename Key, typename Mapped, typename Tag, typename Node_Update, typename Policy_Tl, typename _Alloc>

class `__gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >`

A branched, tree-like (tree, trie) container abstraction.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Tag</i>	Instantiating data structure type, see container_tag.
<i>Node_Update</i>	Updates nodes, restores invariants.
<i>Policy_Tl</i>	Policy typelist.

Template Parameters

<code>_Alloc</code>	Allocator type.
---------------------	-----------------

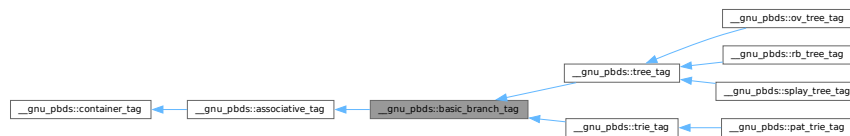
Base is dispatched at compile time via Tag, from the following choices: `tree_tag`, `trie_tag`, and their descendants. Base choices are: `detail::ov_tree_map`, `detail::rb_tree_map`, `detail::splay_tree_map`, and `detail::pat_trie_map`. The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.242 `__gnu_pbds::basic_branch_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::basic_branch_tag`:



6.242.1 Detailed Description

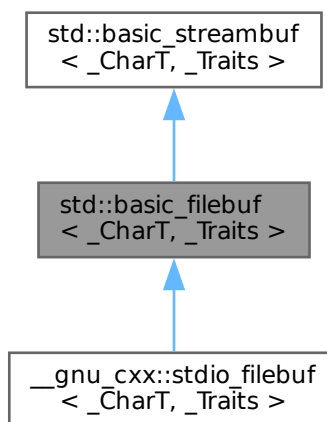
Basic branch structure.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.243 `std::basic_filebuf<_CharT, _Traits>` Class Template Reference

Inheritance diagram for `std::basic_filebuf<_CharT, _Traits>`:



Public Types

- typedef `codecvt`< char_type, char, __state_type > **__codecvt_type**
- typedef `__basic_file`< char > **__file_type**
- typedef `basic_filebuf`< char_type, traits_type > **__filebuf_type**
- typedef traits_type::state_type **__state_type**
- typedef `basic_streambuf`< char_type, traits_type > **__streambuf_type**
- typedef `_CharT` **char_type**
- typedef traits_type::int_type **int_type**
- typedef traits_type::off_type **off_type**
- typedef traits_type::pos_type **pos_type**
- typedef `_Traits` **traits_type**

Public Member Functions

- `basic_filebuf` ()
- `basic_filebuf` (`basic_filebuf` &&)
- `basic_filebuf` (const `basic_filebuf` &)=delete
- virtual `~basic_filebuf` ()
- `__filebuf_type` * `close` ()
- `locale` `getloc` () const
- `streamsize` `in_avail` ()
- bool `is_open` () const throw ()
- template<typename `_Path` >
`_If_fs_path`< `_Path`, `__filebuf_type` * > `open` (const `_Path` &__s, `ios_base::openmode` __mode)
- `__filebuf_type` * `open` (const char *__s, `ios_base::openmode` __mode)
- `__filebuf_type` * `open` (const `std::string` &__s, `ios_base::openmode` __mode)
- `basic_filebuf` & **operator=** (`basic_filebuf` &&)
- `basic_filebuf` & **operator=** (const `basic_filebuf` &)=delete
- `locale` `pubimbue` (const `locale` &__loc)
- int_type `sbumpc` ()
- int_type `sgetc` ()
- `streamsize` `sgetn` (char_type *__s, `streamsize` __n)
- int_type `snextc` ()
- int_type `sputbackc` (char_type __c)
- int_type `sputc` (char_type __c)
- `streamsize` `sputn` (const char_type *__s, `streamsize` __n)
- int_type `sungetc` ()
- void **swap** (`basic_filebuf` &)
- `basic_streambuf` * `pubsetbuf` (char_type *__s, `streamsize` __n)
- pos_type `pubseekoff` (off_type __off, `ios_base::seekdir` __way, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- pos_type `pubseekpos` (pos_type __sp, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- int `pubsync` ()

Protected Member Functions

- void **__safe_gbump** (streamsize __n)
 - void **__safe_pbump** (streamsize __n)
 - void **_M_allocate_internal_buffer** ()
 - bool **_M_convert_to_external** (char_type *, streamsize)
 - void **_M_create_pback** ()
 - void **_M_destroy_internal_buffer** () throw ()
 - void **_M_destroy_pback** () throw ()
 - int **_M_get_ext_pos** (__state_type &__state)
 - pos_type **_M_seek** (off_type __off, ios_base::seekdir __way, __state_type __state)
 - void **_M_set_buffer** (streamsize __off)
 - bool **_M_terminate_output** ()
 - void **gbump** (int __n)
 - virtual void **imbue** (const locale &__loc)
 - virtual int_type **overflow** (int_type __c=_Traits::eof())
 - virtual int_type **pbackfail** (int_type __c=_Traits::eof())
 - void **pbump** (int __n)
 - virtual pos_type **seekoff** (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
 - virtual pos_type **seekpos** (pos_type __pos, ios_base::openmode __mode=ios_base::in|ios_base::out)
 - virtual **__streambuf_type** * **setbuf** (char_type *__s, streamsize __n)
 - void **setg** (char_type *__gbeg, char_type *__gnext, char_type *__gend)
 - void **setp** (char_type *__pbeg, char_type *__pend)
 - virtual **streamsize showmanyc** ()
 - void **swap** (basic_streambuf &__sb)
 - virtual int **sync** ()
 - virtual int_type **uflow** ()
 - virtual int_type **underflow** ()
 - virtual **streamsize xsgetn** (char_type *__s, streamsize __n)
 - virtual **streamsize xsputn** (const char_type *__s, streamsize __n)
-
- char_type * **eback** () const
 - char_type * **gptr** () const
 - char_type * **egptr** () const
-
- char_type * **pbase** () const
 - char_type * **pptr** () const
 - char_type * **epptr** () const

Protected Attributes

- char_type * **_M_buf**
- bool **_M_buf_allocated**
- locale **_M_buf_locale**
- size_t **_M_buf_size**
- const **__codecvt_type** * **_M_codecvt**
- char * **_M_ext_buf**
- **streamsize _M_ext_buf_size**
- char * **_M_ext_end**
- const char * **_M_ext_next**

- `__file_type` **`_M_file`**
 - `char_type` * `_M_in_beg`
 - `char_type` * `_M_in_cur`
 - `char_type` * `_M_in_end`
 - `__c_lock` **`_M_lock`**
 - `ios_base::openmode` `_M_mode`
 - `char_type` * `_M_out_beg`
 - `char_type` * `_M_out_cur`
 - `char_type` * `_M_out_end`
 - `bool` `_M_reading`
 - `__state_type` **`_M_state_beg`**
 - `__state_type` **`_M_state_cur`**
 - `__state_type` **`_M_state_last`**
 - `bool` **`_M_writing`**
-
- `char_type` `_M_pback`
 - `char_type` * `_M_pback_cur_save`
 - `char_type` * `_M_pback_end_save`
 - `bool` `_M_pback_init`

Friends

- class `ios_base`

6.243.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_filebuf< _CharT, _Traits >
```

The actual work of input and output (for files).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class associates both its input and output sequence with an external disk file, and maintains a joint file position for both sequences. Many of its semantics are described in terms of similar behavior in the Standard C Library's `FILE` streams.

Requirements on `traits_type`, specific to this class:

- `traits_type::pos_type` must be `fpos<traits_type::state_type>`
- `traits_type::off_type` must be `streamoff`
- `traits_type::state_type` must be Assignable and DefaultConstructible,
- `traits_type::state_type()` must be the initial state for `codecvt`.

6.243.2 Constructor & Destructor Documentation

`basic_filebuf()`

```
template<typename _CharT , typename _Traits >
std::basic_filebuf< _CharT, _Traits >::basic_filebuf
```

Does not open any files.

The default constructor initializes the parent class using its own default ctor.

References [std::basic_streambuf<_CharT, _Traits>::_M_buf_locale](#).

~basic_filebuf()

```
template<typename _CharT , typename _Traits >
virtual std::basic_filebuf< _CharT, _Traits >::~~basic_filebuf ( ) [inline], [virtual]
```

The destructor closes the file first.

6.243.3 Member Function Documentation

_M_create_pback()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_create_pback ( ) [inline], [protected]
```

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

_M_destroy_pback()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_destroy_pback ( ) throw ( ) [inline], [protected]
```

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

_M_set_buffer()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_set_buffer (
    streamsize __off ) [inline], [protected]
```

This function sets the pointers of the internal buffer, both get and put areas. Typically:

`__off == egptr() - eback()` upon underflow/uflow (**read** mode); `__off == 0` upon overflow (**write** mode); `__off == -1` upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: `eptr() - pbase() == _M_buf_size - 1`, since `_M_buf_size` reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

Referenced by [std::basic_filebuf<_CharT, _Traits>::close\(\)](#).

close()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::__filebuf_type * std::basic_filebuf< _CharT, _Traits >::close
```

Closes the currently associated file.

Returns

`this` on success, NULL on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

References [std::basic_filebuf<_CharT, _Traits>::_M_mode](#), [std::basic_filebuf<_CharT, _Traits>::_M_pback_init](#), [std::basic_filebuf<_CharT, _Traits>::_M_reading](#), and [std::basic_filebuf<_CharT, _Traits>::_M_set_buffer\(\)](#).

eback()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::egptr ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow\(\)](#).

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic\_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
```

```
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
```

```
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow()`.

imbue()

```
template<typename _CharT , typename _Traits >
```

```
void std::basic_filebuf< _CharT, _Traits >::imbue (
    const locale & __loc ) [protected], [virtual]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::ios_base::cur`.

in_avail()

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic_streambuf< _CharT, _Traits >::in_avail ( ) [inline], [inherited]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

is_open()

```
template<typename _CharT , typename _Traits >
bool std::basic_filebuf< _CharT, _Traits >::is_open ( ) const throw ( )    [inline]
```

Returns true if the external file is open.

open() [1/3]

```
template<typename _CharT , typename _Traits >
template<typename _Path >
_If_fs_path< _Path, __filebuf_type * > std::basic_filebuf< _CharT, _Traits >::open (
    const _Path & __s,
    ios_base::openmode __mode )    [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a filesystem::path.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

open() [2/3]

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::__filebuf_type * std::basic_filebuf< _CharT, _Traits >::open (
    const char * __s,
    ios_base::openmode __mode )
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines `app`, `in|out|app`, `in|app`, `binary|app`, `binary|in|out|app`, and `binary|in|app` per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.
A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

References [std::ios_base::app](#), [std::ios_base::cur](#), and [std::ios_base::out](#).

pbackfail()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::int_type std::basic_filebuf< _CharT, _Traits >::pbackfail (
    int_type __c = _Traits::eof() ) [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

eof() on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

References [std::ios_base::cur](#), and [std::ios_base::in](#).

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- epptr() returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline], [inherited]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual `sync` function.

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Referenced by `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< char >::ignore()`, `std::istreambuf_iterator< _CharT, _Traits >::operator++()`, and `std::basic_istream< char >::seekg()`.

seekoff()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf< _CharT, _Traits >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::cur`.

seekpos()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf< _CharT, _Traits >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::beg`.

setbuf()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::__streambuf_type * std::basic_filebuf< _CharT, _Traits >↵
::setbuf (
    char_type * __s,
    streamsize __n ) [protected], [virtual]
```

Manipulates the buffer.

Parameters

<code>↵ __s</code>	Pointer to a buffer area.
<code>↵ __n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

setg()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

setp()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setp (
    char_type * __pbeg,
    char_type * __pend ) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::istreambuf_iterator<_CharT, _Traits>::operator++\(\)](#), and [std::basic_istream<char>::seekg\(\)](#).

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s,__n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_filebuf< _CharT, _Traits >::showmanyc [protected], [virtual]
Investigating the data available.
```

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::ios_base::binary`, and `std::ios_base::in`.

snextc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::snextc ( ) [inline], [inherited]
Getting the next character.
```

Returns

The next character, or eof.

Calls `sputc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::seekg()`, and `std::basic_istream< char >::unget()`.

sputbackc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sputbackc (
    char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::sputn (
    const char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xputn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

sungetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sungetc ( ) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbckfail()`. The effect is to *unget* the last character *gotten*.

Referenced by [std::basic_istream< _CharT, _Traits >::unget\(\)](#).

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_filebuf< _CharT, _Traits >::sync [protected], [virtual]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

uflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual],
[inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#).

underflow()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::int_type std::basic_filebuf< _CharT, _Traits >::underflow
[protected], [virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::ios_base::in`.

xsgetn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_filebuf< _CharT, _Traits >::xsgetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::ios_base::in`.

xspn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_filebuf< _CharT, _Traits >::xspn (
    const char_type * __s,
    streamsize __n ) [protected], [virtual]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

References [std::ios_base::app](#), [std::min\(\)](#), and [std::ios_base::out](#).

6.243.4 Member Data Documentation

M_buf

```
template<typename _CharT , typename _Traits >
char_type* std::basic_filebuf< _CharT, _Traits >::_M_buf [protected]
Pointer to the beginning of internal buffer.
```

M_buf_locale

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale [protected], [inherited]
Current locale setting.
Referenced by std::basic\_filebuf<\_CharT, \_Traits>::basic\_filebuf\(\).
```

M_buf_size

```
template<typename _CharT , typename _Traits >
size_t std::basic_filebuf< _CharT, _Traits >::_M_buf_size [protected]
Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.
```

M_ext_buf

```
template<typename _CharT , typename _Traits >
char* std::basic_filebuf< _CharT, _Traits >::_M_ext_buf [protected]
Buffer for external characters. Used for input when codecvt::always_noconv() == false. When valid, this corresponds to eback().
```

M_ext_buf_size

```
template<typename _CharT , typename _Traits >
streamsize std::basic_filebuf< _CharT, _Traits >::_M_ext_buf_size [protected]
Size of buffer held by _M_ext_buf.
```

M_ext_next

```
template<typename _CharT , typename _Traits >
const char* std::basic_filebuf< _CharT, _Traits >::_M_ext_next [protected]
Pointers into the buffer held by _M_ext_buf that delimit a subsequence of bytes that have been read but not yet converted. When valid, _M_ext_next corresponds to egptr().
```

M_in_beg

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg [protected], [inherited]
Start of get area.
```

`_M_in_cur`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur [protected], [inherited]
Current read area.
```

`_M_in_end`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end [protected], [inherited]
End of get area.
```

`_M_mode`

```
template<typename _CharT , typename _Traits >
ios_base::openmode std::basic_filebuf< _CharT, _Traits >::_M_mode [protected]
Place to stash in || out || in | out settings for current filebuf.
Referenced by std::basic_filebuf< _CharT, _Traits >::close().
```

`_M_out_beg`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg [protected], [inherited]
Start of put area.
```

`_M_out_cur`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur [protected], [inherited]
Current put area.
```

`_M_out_end`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end [protected], [inherited]
End of put area.
```

`_M_pback`

```
template<typename _CharT , typename _Traits >
char_type std::basic_filebuf< _CharT, _Traits >::_M_pback [protected]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

`_M_pback_cur_save`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_filebuf< _CharT, _Traits >::_M_pback_cur_save [protected]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

`_M_pback_end_save`

```
template<typename _CharT , typename _Traits >
char_type* std::basic\_filebuf< _CharT, _Traits >::_M_pback_end_save [protected]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

`_M_pback_init`

```
template<typename _CharT , typename _Traits >
bool std::basic\_filebuf< _CharT, _Traits >::_M_pback_init [protected]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Referenced by [std::basic_filebuf< _CharT, _Traits >::close\(\)](#).

`_M_reading`

```
template<typename _CharT , typename _Traits >
bool std::basic\_filebuf< _CharT, _Traits >::_M_reading [protected]
_M_reading == false && _M_writing == false for uncommitted mode; _M_reading == true for read mode; _M_writing
== true for write mode;
```

NB: `_M_reading == true && _M_writing == true` is unused.

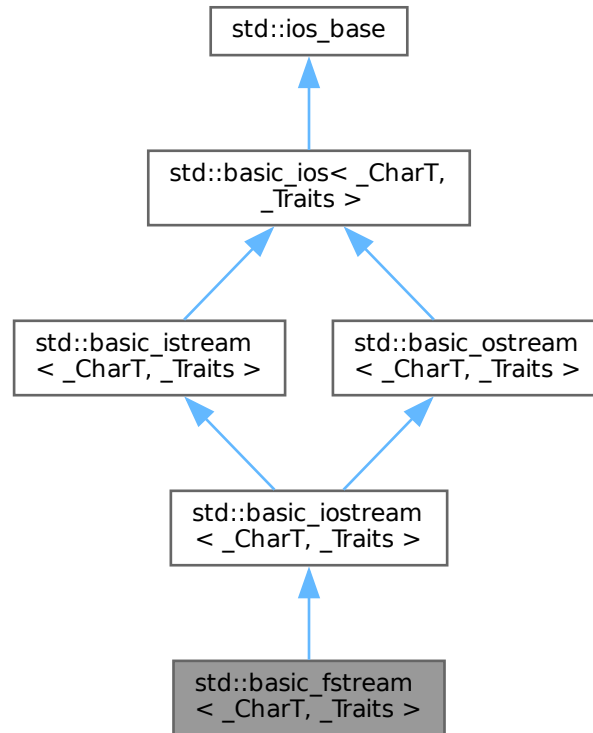
Referenced by [std::basic_filebuf< _CharT, _Traits >::close\(\)](#).

The documentation for this class was generated from the following files:

- [fstream](#)
- [fstream.tcc](#)

6.244 std::basic_fstream< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::basic_fstream< _CharT, _Traits >:



Public Types

- typedef `ctype< _CharT > __ctype_type`
- typedef `ctype< _CharT > __ctype_type`
- typedef `basic_filebuf< char_type, traits_type > __filebuf_type`
- typedef `basic_ios< char_type, traits_type > __ios_type`
- typedef `basic_iostream< char_type, traits_type > __iostream_type`
- typedef `basic_istream< _CharT, _Traits > __istream_type`
- typedef `num_get< _CharT, istreambuf_iterator< _CharT, _Traits > > __num_get_type`
- typedef `num_put< _CharT, ostreambuf_iterator< _CharT, _Traits > > __num_put_type`
- typedef `basic_ostream< _CharT, _Traits > __ostream_type`
- typedef `basic_streambuf< _CharT, _Traits > __streambuf_type`
- typedef `basic_streambuf< _CharT, _Traits > __streambuf_type`
- typedef `_CharT char_type`
- enum `event { erase_event, imbue_event, copyfmt_event }`
- typedef `void(* event_callback)(event __e, ios_base & __b, int __i)`
- typedef `_ios_Fmtflags fmtflags`
- typedef `traits_type::int_type int_type`

- typedef `_ios_istate` `istate`
 - typedef `traits_type::off_type` `off_type`
 - typedef `_ios_Openmode` `openmode`
 - typedef `traits_type::pos_type` `pos_type`
 - typedef `_ios_Seekdir` `seekdir`
 - typedef `_Traits` `traits_type`
-
- typedef `num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>> __num_put_type`

Public Member Functions

- `basic_fstream` ()
- `basic_fstream` (`basic_fstream` &&__rhs)
- template<typename `_Path` , typename `_Require` = `_If_fs_path<_Path>`>>
`basic_fstream` (const `_Path` &__s, `ios_base::openmode` __mode=`ios_base::in|ios_base::out`)
- `basic_fstream` (const `basic_fstream` &)=delete
- `basic_fstream` (const char * __s, `ios_base::openmode` __mode=`ios_base::in|ios_base::out`)
- `basic_fstream` (const `std::string` &__s, `ios_base::openmode` __mode=`ios_base::in|ios_base::out`)
- `~basic_fstream` ()
- template<typename `_ValueT` >
`basic_istream`< `_CharT`, `_Traits` > & `_M_extract` (`_ValueT` &__v)
- const `locale` & `_M_getloc` () const
- template<typename `_ValueT` >
`basic_ostream`< `_CharT`, `_Traits` > & `_M_insert` (`_ValueT` __v)
- void `_M_setstate` (`istate` __state)
- bool `bad` () const
- void `clear` (`istate` __state=`goodbit`)
- void `close` ()
- `basic_ios` & `copyfmt` (const `basic_ios` &__rhs)
- bool `eof` () const
- `istate exceptions` () const
- void `exceptions` (`istate` __except)
- bool `fail` () const
- char_type `fill` () const
- char_type `fill` (char_type __ch)
- `fmtflags flags` () const
- `fmtflags flags` (`fmtflags` __fmtfl)
- `__ostream_type` & `flush` ()
- `streamsize gcount` () const
- `basic_istream`< char > & `getline` (char_type * __s, `streamsize` __n, char_type __delim)
- `basic_istream`< wchar_t > & `getline` (char_type * __s, `streamsize` __n, char_type __delim)
- `locale getloc` () const
- bool `good` () const
- `basic_istream`< char > & `ignore` (`streamsize` __n)
- `basic_istream`< wchar_t > & `ignore` (`streamsize` __n)
- `basic_istream`< char > & `ignore` (`streamsize` __n, int_type __delim)
- `basic_istream`< wchar_t > & `ignore` (`streamsize` __n, int_type __delim)
- `locale imbue` (const `locale` &__loc)
- bool `is_open` ()
- bool `is_open` () const
- long & `isword` (int __ix)

- char [narrow](#) (char_type __c, char __dfault) const
 - template<typename _Path >
_If_fs_path< _Path, void > [open](#) (const _Path &__s, ios_base::openmode __mode=ios_base::in|ios_base::out)
 - void [open](#) (const char * __s, ios_base::openmode __mode=ios_base::in|ios_base::out)
 - void [open](#) (const std::string &__s, ios_base::openmode __mode=ios_base::in|ios_base::out)
 - __ostream_type & [operator<<](#) (__streambuf_type * __sb)
 - __ostream_type & [operator<<](#) (const void * __p)
 - __ostream_type & [operator<<](#) (nullptr_t)
 - basic_fstream & [operator=](#) (basic_fstream &&__rhs)
 - basic_fstream & [operator=](#) (const basic_fstream &)=delete
 - __istream_type & [operator>>](#) (__streambuf_type * __sb)
 - __istream_type & [operator>>](#) (void *&__p)
 - [streamsize precision](#) () const
 - [streamsize precision](#) (streamsize __prec)
 - void *& [pword](#) (int __ix)
 - __filebuf_type * [rdbuf](#) () const
 - basic_streambuf<_CharT, _Traits> * [rdbuf](#) (basic_streambuf<_CharT, _Traits> * __sb)
 - [iostate rdstate](#) () const
 - void [register_callback](#) (event_callback __fn, int __index)
 - __ostream_type & [seekp](#) (off_type, ios_base::seekdir)
 - __ostream_type & [seekp](#) (pos_type)
 - [fmtflags setf](#) (fmtflags __fmtfl)
 - [fmtflags setf](#) (fmtflags __fmtfl, fmtflags __mask)
 - void [setstate](#) (iostate __state)
 - void [swap](#) (basic_fstream &__rhs)
 - pos_type [tellp](#) ()
 - basic_ostream<_CharT, _Traits> * [tie](#) () const
 - basic_ostream<_CharT, _Traits> * [tie](#) (basic_ostream<_CharT, _Traits> * __tiestr)
 - void [unsetf](#) (fmtflags __mask)
 - char_type [widen](#) (char __c) const
 - [streamsize width](#) () const
 - [streamsize width](#) (streamsize __wide)
-
- __istream_type & [operator>>](#) (__istream_type &(*__pf)(__istream_type &))
 - __istream_type & [operator>>](#) (__ios_type &(*__pf)(__ios_type &))
 - __istream_type & [operator>>](#) (ios_base &(*__pf)(ios_base &))

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to false. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- __istream_type & [operator>>](#) (bool &__n)
- __istream_type & [operator>>](#) (short &__n)
- __istream_type & [operator>>](#) (unsigned short &__n)
- __istream_type & [operator>>](#) (int &__n)
- __istream_type & [operator>>](#) (unsigned int &__n)

- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`
- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to true. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type * __s, streamsize __n)`
- `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
- `__istream_type & get (__streambuf_type &__sb)`
- `__istream_type & getline (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type * __s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type * __s, streamsize __n)`
- `streamsize readsome (char_type * __s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`
- `operator bool () const`
- `bool operator! () const`

- `__ostream_type & operator<< (__ostream_type &(__pf)(__ostream_type &))`
- `__ostream_type & operator<< (__ios_type &(__pf)(__ios_type &))`
- `__ostream_type & operator<< (ios_base &(__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
 - `__ostream_type & operator<< (unsigned long __n)`
 - `__ostream_type & operator<< (bool __n)`
 - `__ostream_type & operator<< (short __n)`
 - `__ostream_type & operator<< (unsigned short __n)`
 - `__ostream_type & operator<< (int __n)`
 - `__ostream_type & operator<< (unsigned int __n)`
 - `__ostream_type & operator<< (long long __n)`
 - `__ostream_type & operator<< (unsigned long long __n)`
-
- `__ostream_type & operator<< (double __f)`
 - `__ostream_type & operator<< (float __f)`
 - `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `__ostream_type & write (const char_type * __s, streamsize __n)`

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const `openmode` `__noreplace`
- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`
- static const `fmtflags` `basefield`
- static const `seekdir` `beg`
- static const `openmode` `binary`
- static const `fmtflags` `boolalpha`
- static const `seekdir` `cur`
- static const `fmtflags` `dec`
- static const `seekdir` `end`
- static const `iosstate` `eofbit`
- static const `iosstate` `failbit`
- static const `fmtflags` `fixed`
- static const `fmtflags` `floatfield`
- static const `iosstate` `goodbit`
- static const `fmtflags` `hex`
- static const `openmode` `in`
- static const `fmtflags` `internal`
- static const `fmtflags` `left`

- static const [fmtflags](#) oct
- static const [openmode](#) out
- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) &__loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename [_ValueT](#) >
[__istream_type](#) & [_M_extract](#) ([_ValueT](#) &__v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename [_ValueT](#) >
[__ostream_type](#) & [_M_insert](#) ([_ValueT](#) __v)
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) &__rhs) noexcept
- void [init](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void [move](#) ([basic_ios](#) &&__rhs)
- void [move](#) ([basic_ios](#) &__rhs)
- void [set_rdbuf](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void [swap](#) ([basic_ios](#) &__rhs) noexcept
- void [swap](#) ([basic_ostream](#) &__rhs)
- void [swap](#) ([basic_istream](#) &__rhs)
- void [swap](#) ([basic_ostream](#) &__rhs)

Protected Attributes

- [_Callback_list](#) * [_M_callbacks](#)
- const [__ctype_type](#) * [_M_ctype](#)
- [iostate](#) [_M_exception](#)
- [char_type](#) [_M_fill](#)
- bool [_M_fill_init](#)
- [fmtflags](#) [_M_flags](#)
- [streamsize](#) [_M_gcount](#)
- [locale](#) [_M_ios_locale](#)
- [_Words](#) [_M_local_word](#) [[_S_local_word_size](#)]
- const [__num_get_type](#) * [_M_num_get](#)
- const [__num_put_type](#) * [_M_num_put](#)
- [streamsize](#) [_M_precision](#)

- [basic_streambuf](#)<_CharT, _Traits> * [_M_streambuf](#)
- [iostate](#) [_M_streambuf_state](#)
- [basic_ostream](#)<_CharT, _Traits> * [_M_tie](#)
- [streamsize](#) [_M_width](#)
- [_Words](#) * [_M_word](#)
- [int](#) [_M_word_size](#)
- [_Words](#) [_M_word_zero](#)

6.244.1 Detailed Description

template<typename [_CharT](#), typename [_Traits](#)>

class [std::basic_fstream](#)<_CharT, _Traits>

Controlling input and output for files.

Template Parameters

_CharT	Type of character stream.
_Traits	Traits for character type, defaults to char_traits <_CharT>.

This class supports reading from and writing to named files, using the inherited functions from [std::basic_istream](#). To control the associated sequence, an instance of [std::basic_filebuf](#) is used, which this page refers to as *sb*.

6.244.2 Member Typedef Documentation

[__num_put_type](#)

template<typename [_CharT](#) , typename [_Traits](#) >

typedef [num_put](#)<_CharT, [ostreambuf_iterator](#)<_CharT, _Traits> > [std::basic_ios](#)<_CharT, _Traits>::[__num_put_type](#) [inherited]

These are non-standard types.

[event_callback](#)

typedef void(* [std::ios_base::event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i) [inherited]

The type of an event callback function.

Parameters

_↵ _e	One of the members of the event enum.
_↵ _b	Reference to the ios_base object.
_↵ _i	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several [ios_base](#) and [basic_ios](#) functions, specifically [imbue\(\)](#), [copyfmt\(\)](#), and [~ios\(\)](#).

[fmtflags](#)

typedef [_Ios_Fmtflags](#) [std::ios_base::fmtflags](#) [inherited]

This is a bitmask type.

`__Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

iostate

```
typedef __Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`__Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.244.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.244.4 Constructor & Destructor Documentation**basic_fstream() [1/4]**

```
template<typename _CharT , typename _Traits >
```

```
std::basic_fstream<_CharT, _Traits>::basic_fstream ( ) [inline]
```

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

basic_fstream() [2/4]

```
template<typename _CharT , typename _Traits >
```

```
std::basic_fstream<_CharT, _Traits>::basic_fstream (
```

```
    const char * __s,
```

```
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [explicit]
```

Create an input/output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

basic_fstream() [3/4]

```
template<typename _CharT , typename _Traits >
std::basic_fstream< _CharT, _Traits >::basic_fstream (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [explicit]
```

Create an input/output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

basic_fstream() [4/4]

```
template<typename _CharT , typename _Traits >
template<typename _Path , typename _Require = _If_fs_path<_Path>>
std::basic_fstream< _CharT, _Traits >::basic_fstream (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Create an input/output file stream.

Parameters

<code>__s</code>	<code>filesystem::path</code> specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

~basic_fstream()

```
template<typename _CharT , typename _Traits >
std::basic_fstream< _CharT, _Traits >::~~basic_fstream ( ) [inline]
```

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

6.244.5 Member Function Documentation**_M_getloc()**

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by [std::money_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get\(\)](#), [std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_date\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_weekday\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_year\(\)](#), [std::num_put< _CharT, _Outlter >::do_put\(\)](#), [std::time_put< _CharT, _Outlter >::do_put\(\)](#), and [std::time_put< _CharT, _Outlter >::put\(\)](#).

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#).

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios< _CharT, _Traits >::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::putb\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ios< _CharT, _Traits >::setsb\(\)](#), and [std::basic_istream< _CharT, _Traits >::unget\(\)](#).

close()

```
template<typename _CharT , typename _Traits >
void std::basic_fstream< _CharT, _Traits >::close ( ) [inline]
```

Close the file.

Calls [std::basic_filebuf::close\(\)](#). If that function fails, [failbit](#) is set in the stream's error state.

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (
    const basic_ios< _CharT, _Traits > & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits >::exceptions()`, `std::basic_ios<_CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits >::eof ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the eofbit is set.

Note that other `iosstate` flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits >::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios<_CharT, _Traits >::exceptions ( ) const [inline], [inherited]
Throwing exceptions on errors.
```

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iosstate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits >::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits >::exceptions (
    iosstate __except ) [inline], [inherited]
Throwing exceptions on errors.
```

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
```

```

std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

std::ifstream f ("/etc/motd");

std::cerr << "Setting badbit\n";
f.setstate (std::ios_base::badbit);

std::cerr << "Setting exception mask\n";
f.exceptions (std::ios_base::badbit);
}

```

References [std::basic_ios<_CharT, _Traits>::clear\(\)](#).

fail()

```

template<typename _CharT , typename _Traits >
bool std::basic\_ios<\_CharT, \_Traits>::fail \( \) const [inline], [inherited]

```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::operator bool\(\)](#), [std::basic_ios<_CharT, _Traits>::operator!\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), and [std::basic_ostream<_CharT, _Traits>::tellp\(\)](#).

fill() [1/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \( \) const [inline], [inherited]

```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::widen\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), and [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

fill() [2/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \(
    char\_type \_\_ch ) [inline], [inherited]

```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::operator<<()`, `std::operator>>()`, `std::operator>>()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, and `std::operator>>()`.

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::flush [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::flush()`.

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::gcount ( ) const [inline], [inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::get (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline], [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim ) [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#), [std::basic_streambuf< _CharT, _Traits >::snextc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::setstate\(\)](#).

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

get() [6/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
------------------	----------------------

Parameters

<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits>::getline (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

ignore() [1/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sbumpc\(\)](#).

ignore() [2/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

ignore() [3/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n,
    int_type __delim ) [inherited]
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A “stop” character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), and [std::ios_base::goodbit](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios< _CharT, _Traits >::basic_ios()`.

is_open()

```
template<typename _CharT , typename _Traits >
bool std::basic_fstream< _CharT, _Traits >::is_open ( ) [inline]
```

Wrapper to test for an open file.

Returns

`rdbuf()->is_open()`

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios<_CharT, _Traits>::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c, default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

open() [1/3]

```
template<typename _CharT , typename _Traits >
template<typename _Path >
_If_fs_path<_Path, void> std::basic_fstream<_CharT, _Traits>::open (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a <code>filesystem::path</code> .
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode)`. If that function fails, `failbit` is set in the stream's error state.

open() [2/3]

```
template<typename _CharT , typename _Traits >
void std::basic_fstream<_CharT, _Traits>::open (
    const char * __s,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```


Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode)`. If that function fails, `failbit` is set in the stream's error state.

open() [3/3]

```
template<typename _CharT , typename _Traits >
void std::basic_fstream< _CharT, _Traits >::open (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode)`. If that function fails, `failbit` is set in the stream's error state.

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator"!()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ios_type &(*)(__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    __ostream_type &(*) (__ostream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `io manip` header.

operator<<() [3/17]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<< (
    __streambuf_type * __sb ) [inherited]
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

operator<<() [4/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [5/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    const void * __p ) [inline], [inherited]
```

Pointer arithmetic inserters.

Parameters

$_p$	A variable of pointer type.
-------	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

$_f$	A variable of builtin floating point type.
-------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    float __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

$_f$	A variable of builtin floating point type.
-------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
```

```
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<< (
    int __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [10/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    long double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [13/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    short __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), and [std::basic_ios< _CharT, _Traits >](#)

operator<<() [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_↔$ $_n$	A variable of builtin integral type.
----------------	--------------------------------------

Returns

$*this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [15/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_↔$ $_n$	A variable of builtin integral type.
----------------	--------------------------------------

Returns

$*this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_↔$ $_n$	A variable of builtin integral type.
----------------	--------------------------------------

Returns

$*this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned short __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator>>() [1/17]

```
template<typename _CharT, typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    __ios_type & (*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [2/17]

```
template<typename _CharT, typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    __istream_type & (*) (__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [3/17]

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    __streambuf_type * __sb ) [inherited]
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

operator>>() [4/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    bool & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow <i>f</i>	A variable of builtin floating point type.
--	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    float & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow <i>f</i>	A variable of builtin floating point type.
--	--

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    int & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.


Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [9/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    unsigned int & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [14/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    unsigned long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [15/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    unsigned long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [16/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
```

```
unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

`operator>>()` [17/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    void *& __p ) [inline], [inherited]
```

Basic arithmetic extractors.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

`peek()`

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::peek (
    void ) [inherited]
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

`precision()` [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

put()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (
    char_type __c ) [inherited]
```

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

*this

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#).

putback()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback (
    char_type __c ) [inherited]
```

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sputbackc()`.

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
__filebuf_type * std::basic_fstream<_CharT, _Traits>::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current `basic_filebuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (
    basic_streambuf<_CharT, _Traits> * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios<_CharT, _Traits>::bad()`, `std::basic_ios<_CharT, _Traits>::eof()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_ios<_CharT, _Traits>::good()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::setstate()`.

read()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

*this

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, and `std::basic_ios<_CharT, _Traits>::rdbuf()`.

readsome()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::readsome (
```



```
char_type * __s,
streamsize __n ) [inherited]
```

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rddbuf() -> in_avail()`, called `A` here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_ios< _CharT, _Traits >::rddbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (
    pos_type __pos) [inherited]
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [1/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (
    off_type __off,
    ios_base::seekdir __dir) [inherited]
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [2/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (
    pos_type __pos ) [inherited]
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References `std::basic_ios< _CharT, _Traits >::clear()`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::getline()`, `std::getline()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void ) [inherited]
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rddbuf()` is a null pointer, returns -1.

Otherwise, calls `rddbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf< _CharT, _Traits >::pubsync()`, `std::basic_ios< _CharT, _Traits >::rddbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void ) [inherited]
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::in`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tellp()

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp [inherited]
```

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tie() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::unget (
    void ) [inherited]
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
```

```
char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by [std::basic_ios<_CharT, _Traits>::fill\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::tr2::operator>>\(\)](#), and [std::basic_ostream< char, _Traits>::put\(\)](#).

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by [std::basic_ios< _CharT, _Traits>::copyfmt\(\)](#), [std::num_put< _CharT, _Outiter>::do_put\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns



The previous value of `width()`.

write()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::write (
    const char_type * __s,
    streamsize __n ) [inherited]
```

Character string insertion.

Parameters

 <code>__s</code>	The array to insert.
 <code>__n</code>	Maximum number of characters to insert.

Returns

*this

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.244.6 Member Data Documentation**M_gcount**

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected], [inherited]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of `left`|`right`|`internal`. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by [std::basic_filebuf< _CharT, _Traits >::open\(\)](#).

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ios< _CharT, _Traits >::bad\(\)](#), [std::basic_ios< _CharT, _Traits >::fail\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< char >::get\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< char >::peek\(\)](#), [std::basic_ostream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of dec|oct|hex. Useful for the 2-arg form of setf.

Referenced by [std::dec\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::hex\(\)](#), and [std::oct\(\)](#).

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```

Request a seek relative to the beginning of the stream.

Referenced by [std::basic_filebuf< _CharT, _Traits >::seekpos\(\)](#).

binary

```
const openmode std::ios_base::binary [static], [inherited]
```

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by [std::basic_filebuf< _CharT, _Traits >::showmanyc\(\)](#).

boolalpha

```
const fmtflags std::ios_base::boolalpha [static], [inherited]
```

Insert/extract bool in alphabetic rather than numeric format.

Referenced by [std::boolalpha\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), and [std::noboolalpha\(\)](#).

cur

```
const seekdir std::ios_base::cur [static], [inherited]
```

Request a seek relative to the current position within the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_stringbuf<_CharT, _Traits, std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

dec

```
const fmtflags std::ios_base::dec [static], [inherited]
```

Converts integer input or generates integer output in decimal base.

Referenced by [std::dec\(\)](#).

end

```
const seekdir std::ios_base::end [static], [inherited]
```

Request a seek relative to the current end of the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

eofbit

```
const iostate std::ios_base::eofbit [static], [inherited]
```

Indicates that an input operation reached the end of an input sequence.

```

Referenced by std::basic_istream<_CharT, _Traits>::sentry::sentry(), std::time_get<_CharT, _InIter>::do_get(),
std::num_get<_CharT, _InIter>::do_get(), std::num_get<_CharT, _InIter>::do_get(), std::num_get<_CharT, _InIter>::do_get(),
std::num_get<_CharT, _InIter>::do_get(), std::time_get<_CharT, _InIter>::do_get_date(), std::time_get<_CharT, _InIter>::do_get_
std::time_get<_CharT, _InIter>::do_get_time(), std::time_get<_CharT, _InIter>::do_get_weekday(), std::time_get<_CharT, _InIter>
std::basic_ios<_CharT, _Traits>::eof(), std::basic_istream<char>::get(), std::basic_istream<_CharT, _Traits>::get(),
std::basic_istream<_CharT, _Traits>::get(), std::time_get<_CharT, _InIter>::get(), std::basic_istream<_CharT, _Traits>::operator>
std::operator>>(), std::basic_istream<_CharT, _Traits>::peek(), std::basic_istream<_CharT, _Traits>::putback(),
std::basic_istream<_CharT, _Traits>::read(), std::basic_istream<_CharT, _Traits>::readsome(), std::basic_istream<char>::seekg()
std::basic_istream<_CharT, _Traits>::seekg(), std::basic_istream<_CharT, _Traits>::seekg(), std::basic_istream<_CharT, _Traits>::
and std::ws().

```

failbit

```
const iostate std::ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<char>::ignore()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<char>::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekp()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by `std::fixed()`, and `std::hexfloat()`.

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _Inlter>::do_get\(\)](#),

[std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#),

[std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _Inlter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#),

[std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#),

[std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::put\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unget\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#),

[std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#),

[std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_filebuf<_CharT, _Traits>::underflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::xsgetn\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by [std::num_put<_CharT, _Outlter>::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::pbackfail\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::tellp\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put<_CharT, _Outiter>::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

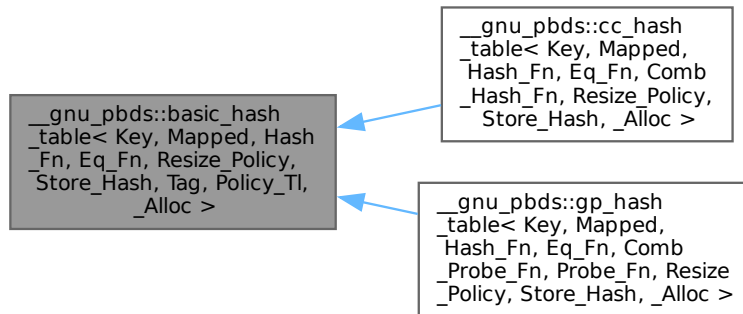
The documentation for this class was generated from the following file:

- [fstream](#)

6.245 __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc > Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >`:



Protected Member Functions

- **basic_hash_table** (const [basic_hash_table](#) &other)
- template<typename T0 >
basic_hash_table (T0 t0)
- template<typename T0 , typename T1 >
basic_hash_table (T0 t0, T1 t1)
- template<typename T0 , typename T1 , typename T2 >
basic_hash_table (T0 t0, T1 t1, T2 t2)
- template<typename T0 , typename T1 , typename T2 , typename T3 >
basic_hash_table (T0 t0, T1 t1, T2 t2, T3 t3)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 >
basic_hash_table (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 >
basic_hash_table (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5)

- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 , typename T7 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6, T7 t7)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 , typename T7 , typename T8 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6, T7 t7, T8 t8)

6.245.1 Detailed Description

`template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename Resize_Policy, bool Store_Hash, typename Tag, typename Policy_Tl, typename _Alloc>`
`class __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >`

A hashed container abstraction.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key.
<i>Tag</i>	Instantiating data structure type, see <code>container_tag</code> .
<i>Policy_Tl</i>	Policy typelist.
<i>_Alloc</i>	Allocator type.

Base is dispatched at compile time via `Tag`, from the following choices: `cc_hash_tag`, `gp_hash_tag`, and descendants of `basic_hash_tag`.

Base choices are: `detail::cc_ht_map`, `detail::gp_ht_map`

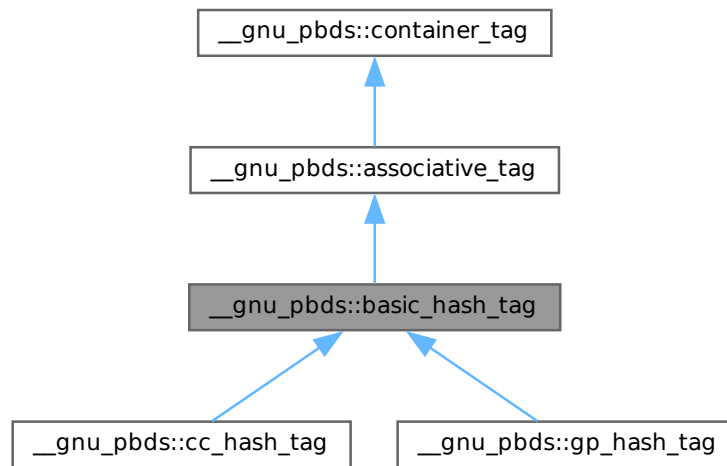
The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.246 `__gnu_pbds::basic_hash_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::basic_hash_tag`:



6.246.1 Detailed Description

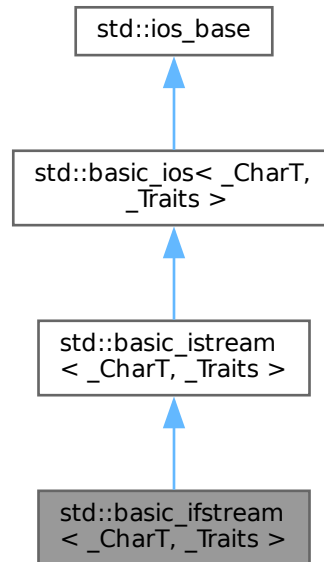
Basic hash structure.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.247 std::basic_ifstream< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::basic_ifstream< _CharT, _Traits >:

**Public Types**

- typedef [ctype](#)< _CharT > **__ctype_type**
- typedef [basic_filebuf](#)< char_type, traits_type > **__filebuf_type**
- typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
- typedef [basic_istream](#)< char_type, traits_type > **__istream_type**
- typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
- typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
- typedef _CharT **char_type**
- enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
- typedef void(* [event_callback](#)) (event __e, [ios_base](#) & __b, int __i)
- typedef _ios_Fmtflags [fmtflags](#)
- typedef traits_type::int_type **int_type**
- typedef _ios_istate [iostate](#)
- typedef traits_type::off_type **off_type**
- typedef _ios_Openmode [openmode](#)
- typedef traits_type::pos_type **pos_type**
- typedef _ios_Seekdir [seekdir](#)
- typedef _Traits **traits_type**

- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_ifstream](#) ()
- **basic_ifstream** ([basic_ifstream](#) &&__rhs)
- template<typename _Path, typename _Require = _If_fs_path<_Path>>
 [basic_ifstream](#) (const _Path &__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- **basic_ifstream** (const [basic_ifstream](#) &)=delete
- [basic_ifstream](#) (const char *__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- [basic_ifstream](#) (const [std::string](#) &__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- [~basic_ifstream](#) ()
- template<typename _ValueT >
 [basic_istream](#)<_CharT, _Traits> & **_M_extract** (_ValueT &__v)
- const [locale](#) & **_M_getloc** () const
- void **_M_setstate** ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- void [close](#) ()
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [iostate exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- char_type [fill](#) () const
- char_type [fill](#) (char_type __ch)
- [fmtflags flags](#) () const
- [fmtflags flags](#) ([fmtflags](#) __fmtfl)
- [streamsize gcount](#) () const
- [basic_istream](#)<char> & [getline](#) (char_type *__s, [streamsize](#) __n, char_type __delim)
- [basic_istream](#)<wchar_t> & [getline](#) (char_type *__s, [streamsize](#) __n, char_type __delim)
- [locale getloc](#) () const
- bool [good](#) () const
- [basic_istream](#)<char> & **ignore** ([streamsize](#) __n)
- [basic_istream](#)<wchar_t> & **ignore** ([streamsize](#) __n)
- [basic_istream](#)<char> & **ignore** ([streamsize](#) __n, int_type __delim)
- [basic_istream](#)<wchar_t> & **ignore** ([streamsize](#) __n, int_type __delim)
- [locale imbue](#) (const [locale](#) &__loc)
- bool [is_open](#) ()
- bool **is_open** () const
- long & [iword](#) (int __ix)
- char [narrow](#) (char_type __c, char __default) const
- template<typename _Path >
 _If_fs_path<_Path, void> [open](#) (const _Path &__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- void [open](#) (const char *__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- void [open](#) (const [std::string](#) &__s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- [basic_ifstream](#) & **operator=** ([basic_ifstream](#) &&__rhs)
- [basic_ifstream](#) & **operator=** (const [basic_ifstream](#) &)=delete
- __istream_type & **operator>>** (__streambuf_type *__sb)
- __istream_type & **operator>>** (void *&__p)
- [streamsize precision](#) () const
- [streamsize precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)

- `__filebuf_type * rdbuf () const`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `void swap (basic_ifstream & __rhs)`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > * __tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to false. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `__istream_type & operator>> (bool & __n)`
 - `__istream_type & operator>> (short & __n)`
 - `__istream_type & operator>> (unsigned short & __n)`
 - `__istream_type & operator>> (int & __n)`
 - `__istream_type & operator>> (unsigned int & __n)`
 - `__istream_type & operator>> (long & __n)`
 - `__istream_type & operator>> (unsigned long & __n)`
 - `__istream_type & operator>> (long long & __n)`
 - `__istream_type & operator>> (unsigned long long & __n)`
-
- `__istream_type & operator>> (float & __f)`
 - `__istream_type & operator>> (double & __f)`
 - `__istream_type & operator>> (long double & __f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to true. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type get ()`
 - `__istream_type & get (char_type &__c)`
 - `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
 - `__istream_type & get (char_type *__s, streamsize __n)`
 - `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
 - `__istream_type & get (__streambuf_type &__sb)`
 - `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `__istream_type & getline (char_type *__s, streamsize __n)`
 - `__istream_type & ignore (streamsize __n, int_type __delim)`
 - `__istream_type & ignore (streamsize __n)`
 - `__istream_type & ignore ()`
 - `int_type peek ()`
 - `__istream_type & read (char_type *__s, streamsize __n)`
 - `streamsize readsome (char_type *__s, streamsize __n)`
 - `__istream_type & putback (char_type __c)`
 - `__istream_type & unget ()`
 - `int sync ()`
 - `pos_type tellg ()`
 - `__istream_type & seekg (pos_type)`
 - `__istream_type & seekg (off_type, ios_base::seekdir)`
-
- `operator bool () const`
 - `bool operator! () const`

Static Public Member Functions

- `static bool sync_with_stdio (bool __sync=true)`
- `static int xalloc () throw ()`

Static Public Attributes

- `static const openmode __noreplace`
- `static const fmtflags adjustfield`
- `static const openmode app`
- `static const openmode ate`
- `static const iostate badbit`
- `static const fmtflags basefield`
- `static const seekdir beg`
- `static const openmode binary`
- `static const fmtflags boolalpha`
- `static const seekdir cur`
- `static const fmtflags dec`
- `static const seekdir end`
- `static const iostate eofbit`
- `static const iostate failbit`
- `static const fmtflags fixed`
- `static const fmtflags floatfield`
- `static const iostate goodbit`
- `static const fmtflags hex`
- `static const openmode in`
- `static const fmtflags internal`
- `static const fmtflags left`
- `static const fmtflags oct`
- `static const openmode out`

- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) & __loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename _ValueT >
 [_istream_type](#) & [_M_extract](#) (_ValueT & __v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) & __rhs) noexcept
- void [init](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
- void [move](#) ([basic_ios](#) && __rhs)
- void [move](#) ([basic_ios](#) & __rhs)
- void [set_rdbuf](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
- void [swap](#) ([basic_ios](#) & __rhs) noexcept
- void [swap](#) ([basic_istream](#) & __rhs)

Protected Attributes

- [_Callback_list](#) * [_M_callbacks](#)
- const [__ctype_type](#) * [_M_ctype](#)
- [iostate](#) [_M_exception](#)
- [char_type](#) [_M_fill](#)
- bool [_M_fill_init](#)
- [fmtflags](#) [_M_flags](#)
- [streamsize](#) [_M_gcount](#)
- [locale](#) [_M_ios_locale](#)
- [_Words](#) [_M_local_word](#) [[_S_local_word_size](#)]
- const [__num_get_type](#) * [_M_num_get](#)
- const [__num_put_type](#) * [_M_num_put](#)
- [streamsize](#) [_M_precision](#)
- [basic_streambuf](#)<_CharT, _Traits> * [_M_streambuf](#)
- [iostate](#) [_M_streambuf_state](#)
- [basic_ostream](#)<_CharT, _Traits> * [_M_tie](#)
- [streamsize](#) [_M_width](#)
- [_Words](#) * [_M_word](#)
- int [_M_word_size](#)
- [_Words](#) [_M_word_zero](#)

6.247.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ifstream< _CharT, _Traits >
```

Controlling input for files.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class supports reading from named files, using the inherited functions from `std::basic_istream`. To control the associated sequence, an instance of `std::basic_filebuf` is used, which this page refers to as `sb`.

6.247.2 Member Typedef Documentation

`__num_put_type`

```
template<typename _CharT , typename _Traits >
typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits
>::__num_put_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

`fmtflags`

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`

- left
- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- badbit
- eofbit
- failbit
- goodbit

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- app
- ate
- binary
- in
- out
- trunc

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.247.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.247.4 Constructor & Destructor Documentation**basic_ifstream() [1/4]**

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ifstream< _CharT, _Traits >::basic_ifstream ( ) [inline]
```

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

basic_ifstream() [2/4]

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ifstream< _CharT, _Traits >::basic_ifstream (
    const char * __s,
    ios_base::openmode __mode = ios_base::in ) [inline], [explicit]
```

Create an input file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

`ios_base::in` is automatically included in `__mode`.

basic_ifstream() [3/4]

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ifstream< _CharT, _Traits >::basic_ifstream (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::in ) [inline], [explicit]
```

Create an input file stream.

Parameters

<code>__s</code>	<code>std::string</code> specifying the filename.
------------------	---

Parameters

<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).
---------------------	--

`ios_base::in` is automatically included in `__mode`.

basic_ifstream() [4/4]

```
template<typename _CharT , typename _Traits >
template<typename _Path , typename _Require = _If_fs_path<_Path>>
std::basic_ifstream< _CharT, _Traits >::basic_ifstream (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::in ) [inline]
```

Create an input file stream.

Parameters

<code>__s</code>	filesystem::path specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

`ios_base::in` is automatically included in `__mode`.

~basic_ifstream()

```
template<typename _CharT , typename _Traits >
std::basic_ifstream< _CharT, _Traits >::~~basic_ifstream ( ) [inline]
```

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

6.247.5 Member Function Documentation**_M_getloc()**

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by `std::money_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_date()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::time_get<_CharT, _Inlter>::get()`, and `std::time_put<_CharT, _Outlter>::put()`.

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other `iostate` flags may also be set.

References `std::ios_base::badbit`, and `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`.

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios< _CharT, _Traits >::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::putb\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ios< _CharT, _Traits >::setsbstate\(\)](#), and [std::basic_istream< _CharT, _Traits >::unget\(\)](#).

close()

```
template<typename _CharT , typename _Traits >
void std::basic_ifstream< _CharT, _Traits >::close ( ) [inline]
```

Close the file.

Calls `std::basic_filebuf::close()`. If that function fails, `failbit` is set in the stream's error state.

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (
    const basic_ios< _CharT, _Traits > & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

References [std::__addressof\(\)](#), [std::basic_ios< _CharT, _Traits >::exceptions\(\)](#), [std::basic_ios< _CharT, _Traits >::fill\(\)](#), [std::ios_base::flags\(\)](#), [std::ios_base::getloc\(\)](#), [std::ios_base::precision\(\)](#), [std::basic_ios< _CharT, _Traits >::tie\(\)](#), [std::tie\(\)](#), and [std::ios_base::width\(\)](#).

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::eof ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the `eofbit` is set.

Note that other `iostate` flags may also be set.

References [std::ios_base::eofbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios< _CharT, _Traits >::exceptions ( ) const [inline], [inherited]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline], [inherited]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References `std::basic_ios<_CharT, _Traits>::clear()`.

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

References `std::ios_base::badbit`, `std::ios_base::failbit`, and `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_ios<_CharT, _Traits>::operator bool()`, `std::basic_ios<_CharT, _Traits>::operator!()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_ifstream< _CharT, _Traits >::gcount ( ) const [inline], [inherited]
Character counting.
```

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits >::int_type std::basic_ifstream< _CharT, _Traits >::get (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References `std::basic_ifstream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline], [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

`*this`

Returns `get(__sb, widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits > & std::basic_ifstream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim ) [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References [std::basic_istream< _CharT, _Traits >::_M_gcount](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::sgetc\(\)](#), [std::basic_streambuf< _CharT, _Traits >::snextc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sputc\(\)](#).

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

`*this`

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::_M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

`*this`

Returns `get(__s, __n, widen("\n"))`.

get() [6/6]

```
template<typename _CharT, typename _Traits>
basic_ifstream<_CharT, _Traits> & std::basic_ifstream<_CharT, _Traits>::get (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A “stop” character.

Returns

`*this`

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, `failbit` is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.



References [std::basic_ifstream<_CharT, _Traits>::_M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_ifstream<_CharT, _Traits>::getline (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

String extraction.

Parameters

 <code>__s</code>	A character array in which to store the data.
 <code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, `std::operator>>()`, and `std::ws()`.

good()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::__detail::operator>>()`.

ignore() [1/3]

```
template<typename _CharT , typename _Traits >
```

```
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_streambuf<_CharT, _Traits>::sbumpc()`.

ignore() [2/3]

```
template<typename _CharT , typename _Traits >
```

```
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_streambuf<_CharT, _Traits>::sgetc()`.

ignore() [3/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    streamsize __n,
    int_type __delim ) [inherited]
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References [std::ios_base::imbue\(\)](#).

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by [std::basic_ios<_CharT, _Traits>::basic_ios\(\)](#).

is_open()

```
template<typename _CharT , typename _Traits >
bool std::basic_ifstream<_CharT, _Traits>::is_open ( ) [inline]
Wrapper to test for an open file.
```

Returns

`rdbuf() -> is_open()`

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios<_CharT, _Traits>::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c, default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

open() [1/3]

```
template<typename _CharT , typename _Traits >
template<typename _Path >
_If_fs_path< _Path, void > std::basic_ifstream< _CharT, _Traits >::open (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::in ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a filesystem::path.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s,__mode|in)`. If that function fails, `failbit` is set in the stream's error state.

open() [2/3]

```
template<typename _CharT , typename _Traits >
void std::basic_ifstream< _CharT, _Traits >::open (
    const char * __s,
    ios_base::openmode __mode = ios_base::in ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(s,__mode|in)`. If that function fails, `failbit` is set in the stream's error state.

open() [3/3]

```
template<typename _CharT , typename _Traits >
void std::basic_ifstream< _CharT, _Traits >::open (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::in ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s,__mode|in)`. If that function fails, `failbit` is set in the stream's error state.

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.
References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator"!()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.
References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator>>() [1/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream<_CharT, _Traits>::operator>> (
    __ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.
For more information, see the `io manip` header.

operator>>() [2/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream<_CharT, _Traits>::operator>> (
    __istream_type &(*) (__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.
For more information, see the `io manip` header.

operator>>() [3/17]

```
template<typename _CharT , typename _Traits >
basic_ifstream<_CharT, _Traits> & std::basic_ifstream<_CharT, _Traits>::operator>> (
    __streambuf_type * __sb ) [inherited]
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

References [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

operator>>() [4/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    bool & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    float & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits > & std::basic_ifstream< _CharT, _Traits >::operator>> (
    int & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.


Manipulators such as std::ws and std::dec use these functions in constructs like std::cin >> std::ws. For more information, see the iomanip header.

operator>>() [9/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::operator>> (
    long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::operator>> (
    long double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
f	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits > & std::basic_ifstream< _CharT, _Traits >::operator>> (
    short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get<_CharT, _InIter>::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
```


Integer arithmetic extractors.

<code>_↵</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Integer arithmetic extractors.

<code>_↵</code>	A variable of builtin integral type.
<code>_n</code>	

Integer arithmetic extractors.

<code>_↵</code>	A variable of builtin integral type.
<code>_n</code>	

Generated by Doxygen

```
unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [17/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_ifstream< _CharT, _Traits >::operator>> (
    void *& __p ) [inline], [inherited]
```

Basic arithmetic extractors.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

peek()

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits >::int_type std::basic_ifstream< _CharT, _Traits >::peek (
    void ) [inherited]
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References [std::basic_ifstream<_CharT, _Traits>::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#).

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

putback()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback (
    char_type __c ) [inherited]
```

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_streambuf< _CharT, _Traits >::sputbackc()`.

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
__filebuf_type * std::basic_ifstream< _CharT, _Traits >::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current basic_filebuf buffer.

This hides both signatures of std::basic_ios::rdbuf().

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_streambuf< _CharT, _Traits > * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See std::ios_base::iosstate for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios<_CharT, _Traits>::bad()`, `std::basic_ios<_CharT, _Traits>::eof()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_ios<_CharT, _Traits>::good()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

read()

```
template<typename _CharT, typename _Traits>
basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::read (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

References `std::basic_istream< _CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, and `std::basic_ios< _CharT, _Traits>::rdbuf()`.

readsome()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_istream< _CharT, _Traits>::readsome (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if `A == -1`, sets `eofbit` and extracts no characters
- if `A == 0`, extracts no characters

- if $A > 0$, extracts $\min(A, n)$

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

References [std::basic_ifstream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::min\(\)](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits > & std::basic_ifstream< _CharT, _Traits >::seekg (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets `failbit`.

Note

This function first clears `eofbit`. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios< _CharT, _Traits >::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ifstream< _CharT, _Traits > & std::basic_ifstream< _CharT, _Traits >::seekg (
    pos_type __pos ) [inherited]
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf() -> pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#) and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl & mask`. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References `std::basic_ios< _CharT, _Traits >::clear()`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::getline()`, `std::getline()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void ) [inherited]
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rddbuf()` is a null pointer, returns -1.

Otherwise, calls `rddbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf< _CharT, _Traits >::pubsync()`, `std::basic_ios< _CharT, _Traits >::rddbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void ) [inherited]
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::in`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tie() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::unget (
    void ) [inherited]
```

Unextracting the previous character.

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios<_CharT, _Traits>::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios<_CharT, _Traits>::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream<char, _Traits>::put()`.

`width()` [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

`width()` [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

`xalloc()`

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.247.6 Member Data Documentation

`_M_gcount`

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic_istream< _CharT, _Traits >::M_gcount [protected], [inherited]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::get()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< char >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_istream< _CharT, _Traits >::ungget()`, and `std::ws()`.

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, and `std::oct()`.

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```

Request a seek relative to the beginning of the stream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`.

binary

```
const openmode std::ios_base::binary [static], [inherited]
```

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by `std::basic_filebuf<_CharT, _Traits>::showmanyc()`.

boolalpha

```
const fmtflags std::ios_base::boolalpha [static], [inherited]
```

Insert/extract `bool` in alphabetic rather than numeric format.

Referenced by `std::boolalpha()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::noboolalpha()`.

cur

```
const seekdir std::ios_base::cur [static], [inherited]
```

Request a seek relative to the current position within the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_stringbuf<_CharT, _Traits, std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

dec

```
const fmtflags std::ios_base::dec [static], [inherited]
```

Converts integer input or generates integer output in decimal base.

Referenced by `std::dec()`.

end

```
const seekdir std::ios_base::end [static], [inherited]
```

Request a seek relative to the current end of the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

eofbit

```
const iostate std::ios_base::eofbit [static], [inherited]
```

Indicates that an input operation reached the end of an input sequence.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::time_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_date()`, `std::time_get<_CharT, _InIter>::do_get_r`, `std::time_get<_CharT, _InIter>::do_get_time()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, `std::time_get<_CharT, _InIter>::do_get_r`, `std::basic_ios<_CharT, _Traits>::eof()`, `std::basic_istream<char>::get()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<char>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, and `std::ws()`.

failbit

```
const iostate std::ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::time_get< _CharT, _InIter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _InIter >::do_get_year\(\)](#), [std::basic_ios< _CharT, _Traits >::fail\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::time_get< _CharT, _InIter >::get\(\)](#), [std::basic_istream< char >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::read\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), and [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#).

fixed

const [fmtflags](#) std::ios_base::fixed [static], [inherited]

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

const [fmtflags](#) std::ios_base::floatfield [static], [inherited]

A mask of scientific|fixed. Useful for the 2-arg form of [setf](#).

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

const [iostate](#) std::ios_base::goodbit [static], [inherited]

Indicates all is well.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::time_get< _CharT, _InIter >::do_get\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::time_get< _CharT, _InIter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _InIter >::do_get_year\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::time_get< _CharT, _InIter >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_ostream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::unsetg\(\)](#), and [std::ws\(\)](#).

hex

const [fmtflags](#) std::ios_base::hex [static], [inherited]

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), and [std::hex\(\)](#).

in

const [openmode](#) std::ios_base::in [static], [inherited]

Open for input. Default for [ifstream](#) and [fstream](#).

Referenced by [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_filebuf< _CharT, _Traits >::pbackfail\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos\(\)](#), [std::basic_filebuf< _CharT, _Traits >::showmanyc\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_filebuf< _CharT, _Traits >::underflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::xsgetn\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by [std::num_put< _CharT, _OutIter >::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::tellp\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

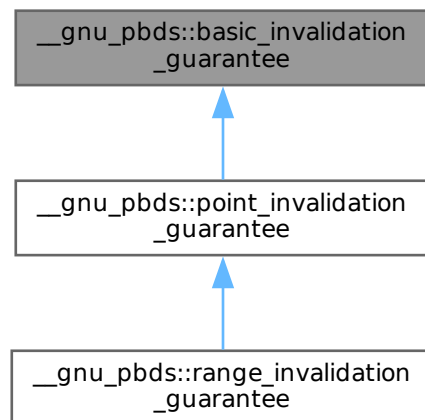
The documentation for this class was generated from the following file:

- [fstream](#)

6.248 `__gnu_pbds::basic_invalidation_guarantee` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::basic_invalidation_guarantee`:



Public Member Functions

- [basic_ios](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
 - virtual [~basic_ios](#) ()
 - const [locale](#) & [_M_getloc](#) () const
 - void [_M_setstate](#) ([iostate](#) __state)
 - bool [bad](#) () const
 - void [clear](#) ([iostate](#) __state=[goodbit](#))
 - [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
 - bool [eof](#) () const
 - [iostate](#) [exceptions](#) () const
 - void [exceptions](#) ([iostate](#) __except)
 - bool [fail](#) () const
 - [char_type](#) [fill](#) () const
 - [char_type](#) [fill](#) ([char_type](#) __ch)
 - [fmtflags](#) [flags](#) () const
 - [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
 - [locale](#) [getloc](#) () const
 - bool [good](#) () const
 - [locale](#) [imbue](#) (const [locale](#) &__loc)
 - long & [iword](#) (int __ix)
 - [char](#) [narrow](#) ([char_type](#) __c, [char](#) __dfault) const
 - [streamsize](#) [precision](#) () const
 - [streamsize](#) [precision](#) ([streamsize](#) __prec)
 - void *& [pword](#) (int __ix)
 - [basic_streambuf](#)< _CharT, _Traits > * [rdbuf](#) () const
 - [basic_streambuf](#)< _CharT, _Traits > * [rdbuf](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
 - [iostate](#) [rdstate](#) () const
 - void [register_callback](#) ([event_callback](#) __fn, int __index)
 - [fmtflags](#) [self](#) ([fmtflags](#) __fmtfl)
 - [fmtflags](#) [self](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
 - void [setstate](#) ([iostate](#) __state)
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) () const
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) ([basic_ostream](#)< _CharT, _Traits > *__tiestr)
 - void [unsetf](#) ([fmtflags](#) __mask)
 - [char_type](#) [widen](#) ([char](#) __c) const
 - [streamsize](#) [width](#) () const
 - [streamsize](#) [width](#) ([streamsize](#) __wide)
-
- [operator bool](#) () const
 - bool [operator!](#) () const

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [openmode](#) `__noreplace`
- static const [fmtflags](#) `adjustfield`
- static const [openmode](#) `app`
- static const [openmode](#) `ate`
- static const [iostate](#) `badbit`
- static const [fmtflags](#) `basefield`
- static const [seekdir](#) `beg`
- static const [openmode](#) `binary`
- static const [fmtflags](#) `boolalpha`
- static const [seekdir](#) `cur`
- static const [fmtflags](#) `dec`
- static const [seekdir](#) `end`
- static const [iostate](#) `eofbit`
- static const [iostate](#) `failbit`
- static const [fmtflags](#) `fixed`
- static const [fmtflags](#) `floatfield`
- static const [iostate](#) `goodbit`
- static const [fmtflags](#) `hex`
- static const [openmode](#) `in`
- static const [fmtflags](#) `internal`
- static const [fmtflags](#) `left`
- static const [fmtflags](#) `oct`
- static const [openmode](#) `out`
- static const [fmtflags](#) `right`
- static const [fmtflags](#) `scientific`
- static const [fmtflags](#) `showbase`
- static const [fmtflags](#) `showpoint`
- static const [fmtflags](#) `showpos`
- static const [fmtflags](#) `skipws`
- static const [openmode](#) `trunc`
- static const [fmtflags](#) `unitbuf`
- static const [fmtflags](#) `uppercase`

Protected Types

- enum { `_S_local_word_size` }

Protected Member Functions

- [basic_ios](#) ()
- [basic_ios](#) (const [basic_ios](#) &)=delete
- void [_M_cache_locale](#) (const [locale](#) &__loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- `_Words` & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) &__rhs) noexcept
- void [init](#) ([basic_streambuf](#)< `_CharT`, `_Traits` > *__sb)
- void [move](#) ([basic_ios](#) &&__rhs)

- void **move** ([basic_ios](#) &__rhs)
- [basic_ios](#) & **operator=** (const [basic_ios](#) &)=delete
- void **set_rdbuf** ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void **swap** ([basic_ios](#) &__rhs) noexcept

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- [char_type](#) **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)< _CharT, _Traits > * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)< _CharT, _Traits > * **_M_tie**
- [streamsize](#) **_M_width**
- [_Words](#) * **_M_word**
- int **_M_word_size**
- [_Words](#) **_M_word_zero**

6.249.1 Detailed Description

template<typename _CharT, typename _Traits>
class std::basic_ios< _CharT, _Traits >

Template class basic_ios, virtual base class for all stream classes.

Template Parameters

_CharT	Type of character stream.
_Traits	Traits for character type, defaults to char_traits<_CharT> .

Most of the member functions called dispatched on stream objects (e.g., `std::cout.foo(bar);`) are consolidated in this class.

6.249.2 Member Typedef Documentation

[__ctype_type](#)

```
template<typename _CharT , typename _Traits >
typedef ctype<_CharT> std::basic\_ios< _CharT, _Traits >::__ctype_type
```

These are non-standard types.

[__num_get_type](#)

```
template<typename _CharT , typename _Traits >
typedef num\_get<_CharT, istreambuf\_iterator<_CharT, _Traits> > std::basic\_ios< _CharT, _Traits >::__num_get_type
```

These are non-standard types.

__num_put_type

```
template<typename _CharT , typename _Traits >
typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits
>::__num_put_type
```

These are non-standard types.

char_type

```
template<typename _CharT , typename _Traits >
typedef _CharT std::basic_ios< _CharT, _Traits >::char_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

event_callback

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the ios_base object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`

- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

int_type

```
template<typename _CharT , typename _Traits >
typedef _Traits::int_type std::basic_ios< _CharT, _Traits >::int_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

off_type

```
template<typename _CharT , typename _Traits >
typedef _Traits::off_type std::basic_ios< _CharT, _Traits >::off_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`

- in
- out
- trunc

pos_type

```
template<typename _CharT , typename _Traits >
typedef _Traits::pos_type std::basic_ios< _CharT, _Traits >::pos_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- beg
- cur, equivalent to `SEEK_CUR` in the C standard library.
- end, equivalent to `SEEK_END` in the C standard library.

traits_type

```
template<typename _CharT , typename _Traits >
typedef _Traits std::basic_ios< _CharT, _Traits >::traits_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

6.249.3 Member Enumeration Documentation

event

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.249.4 Constructor & Destructor Documentation

basic_ios() [1/2]

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::basic_ios (
    basic_streambuf< _CharT, _Traits > * __sb ) [inline], [explicit]
```

Constructor performs initialization.

The parameter is passed by derived streams.

References `std::basic_ios< _CharT, _Traits >::init()`.

~basic_ios()

```
template<typename _CharT , typename _Traits >
virtual std::basic_ios< _CharT, _Traits >::~basic_ios ( ) [inline], [virtual]
```

Empty.

The destructor does nothing. More specifically, it does not destroy the streambuf held by `rdbuf()`.

basic_ios() [2/2]

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::basic_ios ( ) [inline], [protected]
```

Empty.

The default constructor does nothing and is not normally accessible to users.

6.249.5 Member Function Documentation**_M_getloc()**

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Referenced by [std::money_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get\(\)](#), [std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_date\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_weekday\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_year\(\)](#), [std::num_put< _CharT, _Outlter >::do_put\(\)](#), [std::time_put< _CharT, _Outlter >::do_put\(\)](#), [std::time_get< _CharT, _Inlter >::get\(\)](#), and [std::time_put< _CharT, _Outlter >::put\(\)](#).

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#).

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit )
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios< _CharT, _Traits >::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::putb\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ios< _CharT, _Traits >::setsb\(\)](#), and [std::basic_istream< _CharT, _Traits >::unget\(\)](#).

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (
```



```
const basic_ios< _CharT, _Traits > & __rhs )
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References `std::__addressof()`, `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::eof ( ) const [inline]
```

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios< _CharT, _Traits >::exceptions ( ) const [inline]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline]
Fast error checking.
```

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline]
Retrieves the empty character.
```

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline]
Sets a new empty character.
```

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _Outlter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc )
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References [std::ios_base::imbue\(\)](#).

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by [std::basic_ios< _CharT, _Traits >::basic_ios\(\)](#).

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

operator"!"()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#).

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< char >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_ostream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::read\(\)](#), [std::basic_istream< _CharT, _Traits >::readsomewhat\(\)](#), [std::basic_istream< char >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekk\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_streambuf< _CharT, _Traits > * __sb )
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
ios_state std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seek()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::setstate()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

setf() [1/2]

```
fmtflags std::ios_base::setf (
```

```
fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tie() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios< _CharT, _Traits >::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream< char, _Traits >::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.249.6 Member Data Documentation**adjustfield**

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::internal\(\)](#), [std::left\(\)](#), and [std::right\(\)](#).

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by [std::basic_filebuf<_CharT, _Traits>::open\(\)](#).

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::bad\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<char>::get\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_ostream<_CharT, _Traits>::operator<<\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::peek\(\)](#), [std::basic_istream<char>::peek\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_istream<_CharT, _Traits>::ung](#) and [std::ws\(\)](#).

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

`std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, and `std::ws()`.

failbit

```
const iostate std::ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, and `std::basic_ostream< _CharT, _Traits >::seekp()`.

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by `std::fixed()`, and `std::hexfloat()`.

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by `std::defaultfloat()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator<<()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::hex()`.

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by `std::internal()`.

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, and `std::left()`.

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by `std::oct()`.

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekp()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xspn()`.

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by `std::right()`.

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by `std::hexfloat()`, and `std::scientific()`.

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

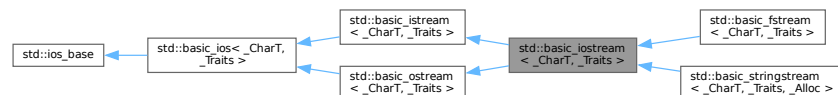
Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [basic_ios.h](#)
- [basic_ios.tcc](#)

6.250 std::basic_istream<_CharT, _Traits> Class Template Reference

Inheritance diagram for `std::basic_istream<_CharT, _Traits>`:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_istream](#)< _CharT, _Traits > **__istream_type**
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
 - typedef [basic_ostream](#)< _CharT, _Traits > **__ostream_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef _Traits::int_type **int_type**
 - typedef _ios_iostate **iostate**
 - typedef _Traits::off_type **off_type**
 - typedef _ios_Openmode **openmode**
 - typedef _Traits::pos_type **pos_type**
 - typedef _ios_Seekdir **seekdir**
 - typedef _Traits **traits_type**
-
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_istream](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- virtual [~basic_istream](#) ()
- template<typename _ValueT >
 [basic_istream](#)< _CharT, _Traits > & **_M_extract** (_ValueT &__v)
- const [locale](#) & **_M_getloc** () const
- template<typename _ValueT >
 [basic_ostream](#)< _CharT, _Traits > & **_M_insert** (_ValueT __v)
- void **_M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & **copyfmt** (const [basic_ios](#) &__rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)
- [fmtflags](#) **flags** () const
- [fmtflags](#) **flags** ([fmtflags](#) __fmtfl)
- **__ostream_type** & **flush** ()
- [streamsize](#) **gcount** () const
- [basic_istream](#)< char > & **getline** (char_type *__s, [streamsize](#) __n, char_type __delim)
- [basic_istream](#)< wchar_t > & **getline** (char_type *__s, [streamsize](#) __n, char_type __delim)

- `locale getloc () const`
 - `bool good () const`
 - `basic_istream< char > & ignore (streamsize __n)`
 - `basic_istream< wchar_t > & ignore (streamsize __n)`
 - `basic_istream< char > & ignore (streamsize __n, int_type __delim)`
 - `basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
 - `locale imbue (const locale &__loc)`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __dfault) const`
 - `__ostream_type & operator<< (__streambuf_type * __sb)`
 - `__ostream_type & operator<< (const void * __p)`
 - `__ostream_type & operator<< (nullptr_t)`
 - `__istream_type & operator>> (__streambuf_type * __sb)`
 - `__istream_type & operator>> (void *& __p)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf () const`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `__ostream_type & seekp (off_type, ios_base::seekdir)`
 - `__ostream_type & seekp (pos_type)`
 - `fmtflags self (fmtflags __fmtfl)`
 - `fmtflags self (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `pos_type tellp ()`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > * __tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `__istream_type & operator>> (bool & __n)`
- `__istream_type & operator>> (short & __n)`
- `__istream_type & operator>> (unsigned short & __n)`
- `__istream_type & operator>> (int & __n)`

- [__istream_type](#) & [operator>>](#) (unsigned int &__n)
- [__istream_type](#) & [operator>>](#) (long &__n)
- [__istream_type](#) & [operator>>](#) (unsigned long &__n)
- [__istream_type](#) & [operator>>](#) (long long &__n)
- [__istream_type](#) & [operator>>](#) (unsigned long long &__n)
- [__istream_type](#) & [operator>>](#) (float &__f)
- [__istream_type](#) & [operator>>](#) (double &__f)
- [__istream_type](#) & [operator>>](#) (long double &__f)

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the [sentry](#) documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type` [get](#) ()
- [__istream_type](#) & [get](#) (char_type &__c)
- [__istream_type](#) & [get](#) (char_type * __s, [streamsize](#) __n, char_type __delim)
- [__istream_type](#) & [get](#) (char_type * __s, [streamsize](#) __n)
- [__istream_type](#) & [get](#) ([__streambuf_type](#) & __sb, char_type __delim)
- [__istream_type](#) & [get](#) ([__streambuf_type](#) & __sb)
- [__istream_type](#) & [getline](#) (char_type * __s, [streamsize](#) __n, char_type __delim)
- [__istream_type](#) & [getline](#) (char_type * __s, [streamsize](#) __n)
- [__istream_type](#) & [ignore](#) ([streamsize](#) __n, int_type __delim)
- [__istream_type](#) & [ignore](#) ([streamsize](#) __n)
- [__istream_type](#) & [ignore](#) ()
- `int_type` [peek](#) ()
- [__istream_type](#) & [read](#) (char_type * __s, [streamsize](#) __n)
- [streamsize](#) [readsome](#) (char_type * __s, [streamsize](#) __n)
- [__istream_type](#) & [putback](#) (char_type __c)
- [__istream_type](#) & [unget](#) ()
- `int` [sync](#) ()
- `pos_type` [tellg](#) ()
- [__istream_type](#) & [seekg](#) (pos_type)
- [__istream_type](#) & [seekg](#) (off_type, [ios_base::seekdir](#))
- [operator bool](#) () const
- `bool` [operator!](#) () const

- [__ostream_type](#) & [operator<<](#) ([__ostream_type](#) &(*__pf)([__ostream_type](#) &))
- [__ostream_type](#) & [operator<<](#) ([__ios_type](#) &(*__pf)([__ios_type](#) &))
- [__ostream_type](#) & [operator<<](#) ([ios_base](#) &(*__pf)([ios_base](#) &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the [sentry](#) documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`
- `__ostream_type & operator<< (short __n)`
- `__ostream_type & operator<< (unsigned short __n)`
- `__ostream_type & operator<< (int __n)`
- `__ostream_type & operator<< (unsigned int __n)`
- `__ostream_type & operator<< (long long __n)`
- `__ostream_type & operator<< (unsigned long long __n)`

- `__ostream_type & operator<< (double __f)`
- `__ostream_type & operator<< (float __f)`
- `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `__ostream_type & write (const char_type * __s, streamsize __n)`

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const `openmode` `__noreplace`
- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`
- static const `fmtflags` `basefield`
- static const `seekdir` `beg`
- static const `openmode` `binary`
- static const `fmtflags` `boolalpha`
- static const `seekdir` `cur`
- static const `fmtflags` `dec`
- static const `seekdir` `end`
- static const `iosstate` `eofbit`
- static const `iosstate` `failbit`
- static const `fmtflags` `fixed`
- static const `fmtflags` `floatfield`
- static const `iosstate` `goodbit`
- static const `fmtflags` `hex`
- static const `openmode` `in`
- static const `fmtflags` `internal`
- static const `fmtflags` `left`

- static const [fmtflags](#) oct
- static const [openmode](#) out
- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- **basic_iostream** ([basic_iostream](#) &&__rhs)
- **basic_iostream** (const [basic_iostream](#) &)=delete
- void [_M_cache_locale](#) (const [locale](#) &__loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename _ValueT >
 [__istream_type](#) & [_M_extract](#) (_ValueT &__v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename _ValueT >
 [__ostream_type](#) & [_M_insert](#) (_ValueT __v)
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) &__rhs) noexcept
- void [init](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void [move](#) ([basic_ios](#) &&__rhs)
- void [move](#) ([basic_ios](#) &__rhs)
- [basic_iostream](#) & [operator=](#) ([basic_iostream](#) &&__rhs)
- [basic_iostream](#) & [operator=](#) (const [basic_iostream](#) &)=delete
- void [set_rdbuf](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void [swap](#) ([basic_ios](#) &__rhs) noexcept
- void [swap](#) ([basic_iostream](#) &__rhs)
- void [swap](#) ([basic_istream](#) &__rhs)
- void [swap](#) ([basic_ostream](#) &__rhs)

Protected Attributes

- [_Callback_list](#) * [_M_callbacks](#)
- const [__ctype_type](#) * [_M_ctype](#)
- [iostate](#) [_M_exception](#)
- [char_type](#) [_M_fill](#)
- bool [_M_fill_init](#)
- [fmtflags](#) [_M_flags](#)
- [streamsize](#) [_M_gcount](#)
- [locale](#) [_M_ios_locale](#)

- `_Words` `_M_local_word` [`_S_local_word_size`]
- `const` `__num_get_type` * `_M_num_get`
- `const` `__num_put_type` * `_M_num_put`
- `streamsize` `_M_precision`
- `basic_streambuf`< `_CharT`, `_Traits` > * `_M_streambuf`
- `iosstate` `_M_streambuf_state`
- `basic_ostream`< `_CharT`, `_Traits` > * `_M_tie`
- `streamsize` `_M_width`
- `_Words` * `_M_word`
- `int` `_M_word_size`
- `_Words` `_M_word_zero`

6.250.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_iostream< _CharT, _Traits >
```

Template class `basic_iostream`.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class multiply inherits from the input and output stream classes simply to provide a single interface.

6.250.2 Member Typedef Documentation

`__num_put_type`

```
template<typename _CharT , typename _Traits >
typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits
>::__num_put_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.250.3 Member Enumeration Documentation

event

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.250.4 Constructor & Destructor Documentation

basic_iostream()

```
template<typename _CharT , typename _Traits >  
std::basic_iostream< _CharT, _Traits >::basic_iostream (  
    basic_streambuf< _CharT, _Traits > * __sb ) [inline], [explicit]
```

Constructor does nothing.

Both of the parent classes are initialized with the same streambuf pointer passed to this constructor.

~basic_iostream()

```
template<typename _CharT , typename _Traits >  
virtual std::basic_iostream< _CharT, _Traits >::~~basic_iostream ( ) [inline], [virtual]
```

Destructor does nothing.

6.250.5 Member Function Documentation

_M_getloc()

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by [std::money_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get\(\)](#), [std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_date\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_weekday\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_year\(\)](#), [std::num_put< _CharT, _Outlter >::do_put\(\)](#), [std::time_get< _CharT, _Inlter >::get\(\)](#), and [std::time_put< _CharT, _Outlter >::put\(\)](#).

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic\_ios< \_CharT, \_Traits >::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other `iostate` flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#).

clear()

```
template<typename _CharT , typename _Traits >
void std::basic\_ios< \_CharT, \_Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios< _CharT, _Traits >::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::putb\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ios< _CharT, _Traits >::setsb\(\)](#), and [std::basic_istream< _CharT, _Traits >::unget\(\)](#).

copyfmt()

```
template<typename _CharT , typename _Traits >
basic\_ios< \_CharT, \_Traits > & std::basic\_ios< \_CharT, \_Traits >::copyfmt (
    const basic\_ios< \_CharT, \_Traits > & \_\_rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits >::exceptions()`, `std::basic_ios<_CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits >::eof ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the `eofbit` is set.

Note that other `iosstate` flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits >::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios<_CharT, _Traits >::exceptions ( ) const [inline], [inherited]
Throwing exceptions on errors.
```

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iosstate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits >::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits >::exceptions (
    iosstate __except ) [inline], [inherited]
Throwing exceptions on errors.
```

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
```

```

std::set_terminate ( __gnu_cxx::__verbose_terminate_handler);

std::ifstream f ( "/etc/motd");

std::cerr << "Setting badbit\n";
f.setstate (std::ios_base::badbit);

std::cerr << "Setting exception mask\n";
f.exceptions (std::ios_base::badbit);
}

```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```

template<typename _CharT , typename _Traits >
bool std::basic\_ios< \_CharT, \_Traits >::fail \( \) const [inline], [inherited]

```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#).

fill() [1/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios< \_CharT, \_Traits >::fill \( \) const [inline], [inherited]

```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios< \_CharT, \_Traits >::fill \(
    char\_type \_\_ch ) [inline], [inherited]

```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::operator<<()`, `std::operator>>()`, `std::operator>>()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, and `std::operator>>()`.

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::flush [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::flush()`.

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::gcount ( ) const [inline], [inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::get (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline], [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

`*this`

Returns `get(__sb,widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim ) [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#), [std::basic_streambuf< _CharT, _Traits >::snextc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::setstate\(\)](#).

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

get() [6/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
------------------	----------------------

Parameters

<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

ignore() [1/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_streambuf< _CharT, _Traits >::sbumpc()`.

ignore() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    streamsize __n ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_streambuf< _CharT, _Traits >::sgetc()`.

ignore() [3/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    streamsize __n,
    int_type __delim ) [inherited]
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A “stop” character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, and `std::ios_base::goodbit`.

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios< _CharT, _Traits >::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c,dfault)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

operator bool()

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

operator"!()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ios_type &(*)(__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ostream_type &(*)(__ostream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [3/17]

```
template<typename _CharT , typename _Traits >
```

```
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    __streambuf_type * __sb ) [inherited]
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

operator<<() [4/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [5/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    const void * __p ) [inline], [inherited]
```

Pointer arithmetic inserters.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    float __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    int __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

↩	A variable of builtin integral type.
<i>n</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [10/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

`operator<<()` [13/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    short __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::num_put< _CharT, _Outiter >::put\(\)](#), and [std::basic_ios< _CharT, _Traits >](#)

`operator<<()` [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.


`operator<<()` [15/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned short __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator>>() [1/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
```

```
__ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [2/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    __istream_type &(*) (__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [3/17]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (
    __streambuf_type * __sb ) [inherited]
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

operator>>() [4/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    bool & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

↵	A variable of builtin floating point type.
_↵	
↵	
_↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    float & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

↵	A variable of builtin floating point type.
_↵	
↵	
_↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    int & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [9/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    long double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

<code>f</code>	A variable of builtin floating point type.
----------------	--

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.


References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned int & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [14/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [15/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [16/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [17/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    void *& __p ) [inline], [inherited]
```

Basic arithmetic extractors.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

peek()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::peek (
    void ) [inherited]
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

put()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (
    char_type __c ) [inherited]
```

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

*this

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#).

putback()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback (
    char_type __c ) [inherited]
```

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.


References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sputbackc\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix )    [inline], [inherited]
```

Access to void pointer array.

Parameters

 <code>__ix</code>	Index into the array.
---	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf<_CharT, _Traits > * std::basic_ios<_CharT, _Traits>::rdbuf ( ) const    [inline],
[inherited]
```

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<char>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::tr2::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::peek\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsomewhat\(\)](#), [std::basic_istream<char>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_ostream<_CharT, _Traits>::tellg\(\)](#), [std::basic_istream<_CharT, _Traits>::unget\(\)](#), and [std::ws\(\)](#).

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf<_CharT, _Traits > * std::basic_ios<_CharT, _Traits>::rdbuf (
    basic_streambuf<_CharT, _Traits > * __sb )    [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

`rdstate()`

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seek()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::tellg()`.

`read()`

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.



References [std::basic_istream<_CharT, _Traits>::_M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), and [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#).

readsome()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_istream<_CharT, _Traits>::readsome (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction until the buffer is exhausted, but no more.

Parameters

 <code>__s</code>	A character array.
 <code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf() -> in_avail()`, called A here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

References [std::basic_istream<_CharT, _Traits>::_M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::min\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios< _CharT, _Traits >::failbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#) and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    pos_type __pos ) [inherited]
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios< _CharT, _Traits >::failbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#) and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

seekp() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (
```

```
    off_type __off,  
    ios_base::seekdir __dir ) [inherited]
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

seekp() [2/2]

```
template<typename _CharT , typename _Traits >  
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (  
    pos_type __pos ) [inherited]
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

setf() [1/2]

```
fmtflags std::ios_base::setf (  
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#),

[std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#),

[std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT,](#)

[std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits](#)

[std::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#),

[std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits](#)

[std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >](#)

[std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void ) [inherited]
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_streambuf<_CharT, _Traits>::pubsync\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void ) [inherited]
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References [std::ios_base::badbit](#), [std::ios_base::cur](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::in](#), and [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#).

tellp()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp [inherited]
```

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tie() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::unget (
    void ) [inherited]
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf() -> sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios<_CharT, _Traits>::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios<_CharT, _Traits>::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream<char, _Traits>::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

write()

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (
    const char_type * __s,
    streamsize __n ) [inherited]
```

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.250.6 Member Data Documentation

M_gcount

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected], [inherited]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::get()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< char >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, and `std::oct()`.

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```


failbit

```
const iostate std::ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<char>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::read\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), and [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unget\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#), and [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#).

`std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by `std::internal()`.

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, and `std::left()`.

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by `std::oct()`.

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by `std::right()`.

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by `std::hexfloat()`, and `std::scientific()`.

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for ofstream.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

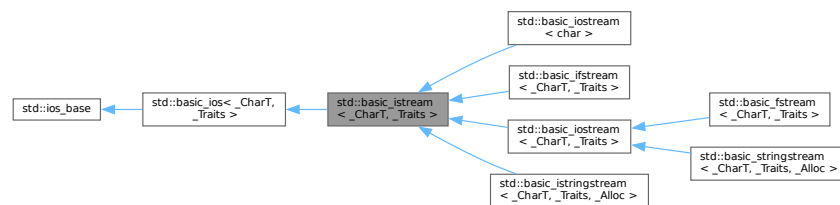
Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [istream](#)

6.251 std::basic_istream<_CharT, _Traits> Class Template Reference

Inheritance diagram for `std::basic_istream<_CharT, _Traits>`:

**Classes**

- class [sentry](#)

Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_istream](#)< _CharT, _Traits > **__istream_type**
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef _Traits::int_type **int_type**
 - typedef _ios_istate **iostate**
 - typedef _Traits::off_type **off_type**
 - typedef _ios_Openmode **openmode**
 - typedef _Traits::pos_type **pos_type**
 - typedef _ios_Seekdir **seekdir**
 - typedef _Traits **traits_type**
-
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_istream](#) ([__streambuf_type](#) *__sb)
- virtual [~basic_istream](#) ()
- template<typename _ValueT >
 [basic_istream](#)< _CharT, _Traits > & **M_extract** (_ValueT &__v)
- const [locale](#) & **M_getloc** () const
- void **M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & **copyfmt** (const [basic_ios](#) &__rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- [char_type](#) **fill** () const
- [char_type](#) **fill** ([char_type](#) __ch)
- [fmtflags](#) **flags** () const
- [fmtflags](#) **flags** ([fmtflags](#) __fmtfl)
- [streamsize](#) **gcount** () const
- [basic_istream](#)< [char](#) > & **getline** ([char_type](#) *__s, [streamsize](#) __n, [char_type](#) __delim)
- [basic_istream](#)< [wchar_t](#) > & **getline** ([char_type](#) *__s, [streamsize](#) __n, [char_type](#) __delim)
- [locale](#) **getloc** () const
- bool **good** () const
- [basic_istream](#)< [char](#) > & **ignore** ([streamsize](#) __n)
- [basic_istream](#)< [wchar_t](#) > & **ignore** ([streamsize](#) __n)
- [basic_istream](#)< [char](#) > & **ignore** ([streamsize](#) __n, [int_type](#) __delim)
- [basic_istream](#)< [wchar_t](#) > & **ignore** ([streamsize](#) __n, [int_type](#) __delim)
- [locale](#) **imbue** (const [locale](#) &__loc)
- long & **inword** (int __ix)

- `char narrow (char_type __c, char __dfault) const`
 - `__istream_type & operator>> (__streambuf_type *__sb)`
 - `__istream_type & operator>> (void *&__p)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf () const`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > *__sb)`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > *__tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `__istream_type & operator>> (bool &__n)`
 - `__istream_type & operator>> (short &__n)`
 - `__istream_type & operator>> (unsigned short &__n)`
 - `__istream_type & operator>> (int &__n)`
 - `__istream_type & operator>> (unsigned int &__n)`
 - `__istream_type & operator>> (long &__n)`
 - `__istream_type & operator>> (unsigned long &__n)`
 - `__istream_type & operator>> (long long &__n)`
 - `__istream_type & operator>> (unsigned long long &__n)`
-
- `__istream_type & operator>> (float &__f)`
 - `__istream_type & operator>> (double &__f)`
 - `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type *__s, streamsize __n)`
- `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
- `__istream_type & get (__streambuf_type &__sb)`
- `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type *__s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type *__s, streamsize __n)`
- `streamsize readsome (char_type *__s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`

- `operator bool () const`
- `bool operator! () const`

Static Public Member Functions

- static `bool sync_with_stdio (bool __sync=true)`
- static `int xalloc () throw ()`

Static Public Attributes

- static const `openmode __noreplace`
- static const `fmtflags adjustfield`
- static const `openmode app`
- static const `openmode ate`
- static const `iosstate badbit`
- static const `fmtflags basefield`
- static const `seekdir beg`
- static const `openmode binary`
- static const `fmtflags boolalpha`
- static const `seekdir cur`
- static const `fmtflags dec`
- static const `seekdir end`
- static const `iosstate eofbit`
- static const `iosstate failbit`
- static const `fmtflags fixed`
- static const `fmtflags floatfield`
- static const `iosstate goodbit`
- static const `fmtflags hex`

- static const [openmode in](#)
- static const [fmtflags internal](#)
- static const [fmtflags left](#)
- static const [fmtflags oct](#)
- static const [openmode out](#)
- static const [fmtflags right](#)
- static const [fmtflags scientific](#)
- static const [fmtflags showbase](#)
- static const [fmtflags showpoint](#)
- static const [fmtflags showpos](#)
- static const [fmtflags skipws](#)
- static const [openmode trunc](#)
- static const [fmtflags unitbuf](#)
- static const [fmtflags uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- **basic_istream** ([basic_istream](#) &&__rhs)
- **basic_istream** (const [basic_istream](#) &)=delete
- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- template<typename _ValueT >
 [__istream_type](#) & **_M_extract** (_ValueT &__v)
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)<_CharT, _Traits > *__sb)
- void **move** ([basic_ios](#) &&__rhs)
- void **move** ([basic_ios](#) &__rhs)
- [basic_istream](#) & **operator=** ([basic_istream](#) &&__rhs)
- [basic_istream](#) & **operator=** (const [basic_istream](#) &)=delete
- void **set_rdbuf** ([basic_streambuf](#)<_CharT, _Traits > *__sb)
- void **swap** ([basic_ios](#) &__rhs) noexcept
- void **swap** ([basic_istream](#) &__rhs)

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- [char_type](#) **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [streamsize](#) **_M_gcount**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]

- const [__num_get_type](#) * [_M_num_get](#)
- const [__num_put_type](#) * [_M_num_put](#)
- [streamsize](#) [_M_precision](#)
- [basic_streambuf](#)<_CharT, _Traits> * [_M_streambuf](#)
- [ios_base](#) [_M_streambuf_state](#)
- [basic_ostream](#)<_CharT, _Traits> * [_M_tie](#)
- [streamsize](#) [_M_width](#)
- [_Words](#) * [_M_word](#)
- int [_M_word_size](#)
- [_Words](#) [_M_word_zero](#)

Friends

- class [sentry](#)

6.251.1 Detailed Description

template<typename _CharT, typename _Traits>
class std::basic_istream<_CharT, _Traits>

Template class basic_istream.

Template Parameters

_CharT	Type of character stream.
_Traits	Traits for character type, defaults to char_traits<_CharT> .

This is the base class for all input streams. It provides text formatting of all builtin types, and communicates with any class derived from basic_streambuf to do the actual input.

6.251.2 Member Typedef Documentation

[__num_put_type](#)

```
template<typename _CharT, typename _Traits>
typedef num\_put<_CharT, ostreambuf\_iterator<_CharT, _Traits> > std::basic\_ios<_CharT, _Traits>::\_\_num\_put\_type [inherited]
```

These are non-standard types.

[event_callback](#)

```
typedef void(* std::ios\_base::event\_callback) (event __e, ios\_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

__e	One of the members of the event enum.
__b	Reference to the ios_base object.
__i	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically

`imbue()`, `copyfmt()`, and `~ios()`.

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.251.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.251.4 Constructor & Destructor Documentation**basic_istream()**

```
template<typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits >::basic_istream (
    __streambuf_type * __sb ) [inline], [explicit]
```

Base constructor.

This ctor is almost never called by the user directly, rather from derived classes' initialization lists, which pass a pointer to their own stream buffer.

~basic_istream()

```
template<typename _CharT , typename _Traits >
virtual std::basic_istream< _CharT, _Traits >::~~basic_istream ( ) [inline], [virtual]
```

Base destructor.

This does very little apart from providing a virtual base dtor.

6.251.5 Member Function Documentation

`_M_getloc()`

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by [std::money_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_date\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_weekday\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_year\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), [std::time_get<_CharT, _Inlter>::get\(\)](#), and [std::time_put<_CharT, _Outlter>::put\(\)](#).

`bad()`

```
template<typename _CharT , typename _Traits >  
bool std::basic\_ios<\_CharT, \_Traits>::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#).

`clear()`

```
template<typename _CharT , typename _Traits >  
void std::basic\_ios<\_CharT, \_Traits>::clear (  
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios<_CharT, _Traits>::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::putb\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ios<_CharT, _Traits>::setsb\(\)](#), and [std::basic_istream<_CharT, _Traits>::unget\(\)](#).

`copyfmt()`

```
template<typename _CharT , typename _Traits >  
basic\_ios<\_CharT, \_Traits> & std::basic\_ios<\_CharT, \_Traits>::copyfmt (  
    const basic\_ios<\_CharT, \_Traits> & \_\_rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References [std::__addressof\(\)](#), [std::basic_ios<_CharT, _Traits>::exceptions\(\)](#), [std::basic_ios<_CharT, _Traits>::fill\(\)](#), [std::ios_base::flags\(\)](#), [std::ios_base::getloc\(\)](#), [std::ios_base::precision\(\)](#), [std::basic_ios<_CharT, _Traits>::tie\(\)](#), [std::tie\(\)](#), and [std::ios_base::width\(\)](#).

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::eof ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::eofbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios<_CharT, _Traits>::exceptions ( ) const [inline], [inherited]
Throwing exceptions on errors.
```

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#).

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits>::exceptions (
    iostate __except ) [inline], [inherited]
Throwing exceptions on errors.
```

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
```

```

std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

std::ifstream f ("/etc/motd");

std::cerr << "Setting badbit\n";
f.setstate (std::ios_base::badbit);

std::cerr << "Setting exception mask\n";
f.exceptions (std::ios_base::badbit);
}

```

References [std::basic_ios<_CharT, _Traits>::clear\(\)](#).

fail()

```

template<typename _CharT , typename _Traits >
bool std::basic\_ios<\_CharT, \_Traits>::fail \( \) const [inline], [inherited]

```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::operator bool\(\)](#), [std::basic_ios<_CharT, _Traits>::operator!\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), and [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

fill() [1/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \( \) const [inline], [inherited]

```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::widen\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), and [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

fill() [2/2]

```

template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \(
    char\_type \_\_ch ) [inline], [inherited]

```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream<_CharT, _Traits>::gcount ( ) const [inline]
Character counting.
```

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::get (
    void )
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim )
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A “stop” character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::get()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, `std::basic_streambuf< _CharT, _Traits >::snexctc()`, and `std::basic_streambuf< _CharT, _Traits >::sbumpc()`.

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c )
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::get (
    char_type * __s,
    streamsize __n ) [inline]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen('\n'))`.

get() [6/6]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::get (
    char_type * __s,
    streamsize __n,
    char_type __delim )
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream<_CharT, _Traits>::_M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::getline (
    char_type * __s,
    streamsize __n ) [inline]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

*this

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim )
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim )
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

ignore() [1/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    void )
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sbumpc\(\)](#).

ignore() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    streamsize __n )
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#).

ignore() [3/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (
    streamsize __n,
    int_type __delim )
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios< _CharT, _Traits >::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

operator bool()

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator"!()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator>>() [1/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    __ios_type &(*)(__ios_type &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.

For more information, see the `io manip` header.

operator>>() [2/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    __istream_type &(*)(__istream_type &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.

For more information, see the `io manip` header.

operator>>() [3/17]

```
template<typename _CharT , typename _Traits >
```

```
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    __streambuf_type * __sb )
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

References [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [4/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    bool & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline]
```

Floating point arithmetic extractors.

Parameters

<code>f</code>	A variable of builtin floating point type.
----------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    float & __f ) [inline]
```

Floating point arithmetic extractors.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    int & __n )
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_ n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [9/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long double & __f ) [inline]
```

Floating point arithmetic extractors.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow <i>f</i>	A variable of builtin floating point type.
--	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    short & __n )
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.


References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned int & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [14/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [15/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long long & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [16/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned short & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [17/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    void *& __p ) [inline]
```

Basic arithmetic extractors.

Parameters

 __p	A variable of pointer type.
--	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

peek()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::peek (
    void )
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

putback()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback (
    char_type __c )
```

Unextracting a single character.

`std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_streambuf< _CharT, _Traits > * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

read()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (
    char_type * __s,
    streamsize __n )
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

*this

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

readsome()

```
template<typename _CharT, typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::readsome (
    char_type * __s,
    streamsize __n )
```

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the `streambuf`'s buffer, `rdbuf() -> in_avail()`, called `A` here:

- if `A == -1`, sets `eofbit` and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the `streambuf`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    off_type __off,
    ios_base::seekdir __dir )
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    pos_type __pos )
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears `eofbit`. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#) and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl & mask`. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits>::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References `std::basic_ios< _CharT, _Traits >::clear()`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::getline()`, `std::getline()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void )
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf< _CharT, _Traits >::pubsync()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void )
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::in`, and `std::basic_ios<_CharT, _Traits>::rdbuf()`.

tie() [1/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_ios<_CharT, _Traits>::copyfmt()`.

tie() [2/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::tie (
    basic_ostream<_CharT, _Traits> * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::unget (
    void )
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios<_CharT, _Traits>::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios<_CharT, _Traits>::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream<char, _Traits>::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of width().

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.251.6 Member Data Documentation**M_gcount**

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by [std::basic_filebuf<_CharT, _Traits>::open\(\)](#).

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::bad\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<char>::get\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_ostream<_CharT, _Traits>::operator<<\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::peek\(\)](#), [std::basic_istream<char>::peek\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_istream<_CharT, _Traits>::ung](#) and [std::ws\(\)](#).

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

Referenced by [std::dec\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::hex\(\)](#), and [std::oct\(\)](#).

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```

Request a seek relative to the beginning of the stream.

Referenced by [std::basic_filebuf<_CharT, _Traits>::seekpos\(\)](#).

binary

```
const openmode std::ios_base::binary [static], [inherited]
```

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#).

boolalpha

```
const fmtflags std::ios_base::boolalpha [static], [inherited]
```

Insert/extract `bool` in alphabetic rather than numeric format.

Referenced by [std::boolalpha\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::noboolalpha\(\)](#).

cur

```
const seekdir std::ios_base::cur [static], [inherited]
```

Request a seek relative to the current position within the sequence.

Referenced by [std::basic_filebuf<_CharT, _Traits>::imbue\(\)](#), [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_filebuf<_CharT, _Traits>::seekoff\(\)](#), [std::basic_stringbuf<_CharT, _Traits>::seekoff\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), and [std::basic_ostream<_CharT, _Traits>::tellp\(\)](#).

dec

```
const fmtflags std::ios_base::dec [static], [inherited]
```

Converts integer input or generates integer output in decimal base.

Referenced by [std::dec\(\)](#).

end

```
const seekdir std::ios_base::end [static], [inherited]
```

Request a seek relative to the current end of the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

eofbit

```
const iostate std::ios_base::eofbit [static], [inherited]
```

Indicates that an input operation reached the end of an input sequence.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_date\(\)](#), [std::time_get<_CharT, _InIter>::do_get_r](#), [std::time_get<_CharT, _InIter>::do_get_time\(\)](#), [std::time_get<_CharT, _InIter>::do_get_weekday\(\)](#), [std::time_get<_CharT, _InIter>](#), [std::basic_ios<_CharT, _Traits>::eof\(\)](#), [std::basic_istream<char>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>](#), [std::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::peek\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::read\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<char>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::](#) and [std::ws\(\)](#).

failbit

```
const iostate std::ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, and `std::basic_ostream< _CharT, _Traits >::seekp()`.

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```


Indicates all is well.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _Inlter>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<char, _Traits>::operator<<()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<char>::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::unset()`, and `std::ws()`.

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by `std::num_get<_CharT, _Inlter>::do_get()`, `std::num_put<_CharT, _Outlter>::do_put()`, and `std::hex()`.

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by `std::internal()`.

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by `std::num_put<_CharT, _Outlter>::do_put()`, and `std::left()`.

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by `std::oct()`.

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::pbackfail()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos()`.

`std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by `std::right()`.

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by `std::hexfloat()`, and `std::scientific()`.

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by `std::noshowpos()`, and `std::showpos()`.

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::noskipws()`, `std::__detail::operator>>()`, and `std::skipws()`.

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by `std::nounitbuf()`, and `std::unitbuf()`.

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

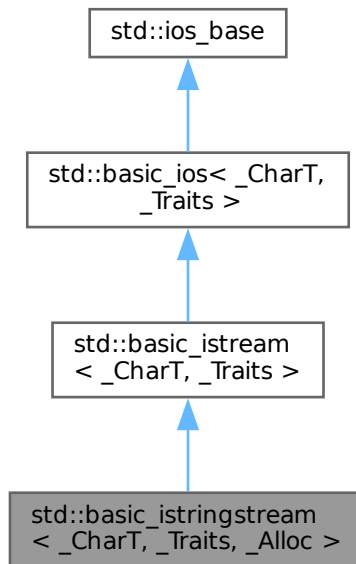
Replaces certain lowercase letters with their uppercase equivalents in generated output.

Referenced by [std::num_put<_CharT, _OutIter >::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).
 The documentation for this class was generated from the following files:

- [iosfwd](#)
- [istream](#)
- [istream.tcc](#)

6.252 std::basic_istream<_CharT, _Traits, _Alloc > Class Template Reference

Inheritance diagram for `std::basic_istream<_CharT, _Traits, _Alloc >`:



Public Types

- typedef `ctype<_CharT>` `__ctype_type`
- typedef `basic_ios<_CharT, _Traits>` `__ios_type`
- typedef `basic_istream<char_type, traits_type>` `__istream_type`
- typedef `num_get<_CharT, istreambuf_iterator<_CharT, _Traits>>` `__num_get_type`
- typedef `basic_streambuf<_CharT, _Traits>` `__streambuf_type`
- typedef `basic_string<_CharT, _Traits, _Alloc>` `__string_type`
- typedef `basic_stringbuf<_CharT, _Traits, _Alloc>` `__stringbuf_type`
- typedef `_Alloc` `allocator_type`
- typedef `_CharT` `char_type`
- enum `event` { `erase_event`, `imbue_event`, `copyfmt_event` }
- typedef `void(* event_callback)(event __e, ios_base &__b, int __i)`
- typedef `_ios_Fmtflags` `fmtflags`
- typedef `traits_type::int_type` `int_type`
- typedef `_ios_istate` `istate`

- typedef traits_type::off_type **off_type**
- typedef _ios_Openmode **openmode**
- typedef traits_type::pos_type **pos_type**
- typedef _ios_Seekdir **seekdir**
- typedef _Traits **traits_type**
- typedef num_put< _CharT, ostreambuf_iterator< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- **basic_istream** ()
- **basic_istream** (basic_istream &&__rhs)
- **basic_istream** (const __string_type &__str, ios_base::openmode __mode=ios_base::in)
- **basic_istream** (const basic_istream &)=delete
- **basic_istream** (ios_base::openmode __mode)
- ~basic_istream ()
- template<typename _ValueT >
 basic_istream< _CharT, _Traits > & **_M_extract** (_ValueT &__v)
- const locale & **_M_getloc** () const
- void **_M_setstate** (iostate __state)
- bool **bad** () const
- void **clear** (iostate __state=goodbit)
- basic_ios & **copyfmt** (const basic_ios &__rhs)
- bool **eof** () const
- iostate **exceptions** () const
- void **exceptions** (iostate __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)
- fmtflags **flags** () const
- fmtflags **flags** (fmtflags __fmtfl)
- streamsize **gcount** () const
- basic_istream< char > & **getline** (char_type * __s, streamsize __n, char_type __delim)
- basic_istream< wchar_t > & **getline** (char_type * __s, streamsize __n, char_type __delim)
- locale **getloc** () const
- bool **good** () const
- basic_istream< char > & **ignore** (streamsize __n)
- basic_istream< wchar_t > & **ignore** (streamsize __n)
- basic_istream< char > & **ignore** (streamsize __n, int_type __delim)
- basic_istream< wchar_t > & **ignore** (streamsize __n, int_type __delim)
- locale **imbue** (const locale &__loc)
- long & **inword** (int __ix)
- char **narrow** (char_type __c, char __dfault) const
- basic_istream & **operator=** (basic_istream &&__rhs)
- basic_istream & **operator=** (const basic_istream &)=delete
- __istream_type & **operator>>** (__streambuf_type * __sb)
- __istream_type & **operator>>** (void *& __p)
- streamsize **precision** () const
- streamsize **precision** (streamsize __prec)
- void *& **pword** (int __ix)
- __stringbuf_type * **rdbuf** () const

- `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
- `iosate rdstate () const`
- `void register_callback (event_callback __fn, int __index)`
- `fmtflags setf (fmtflags __fmtfl)`
- `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
- `void setstate (iosate __state)`
- `__string_type str () const`
- `void str (const __string_type & __s)`
- `void swap (basic_istream & __rhs)`
- `basic_ostream< _CharT, _Traits > * tie () const`
- `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > * __tiestr)`
- `void unsetf (fmtflags __mask)`
- `char_type widen (char __c) const`
- `streamsize width () const`
- `streamsize width (streamsize __wide)`
- `__istream_type & operator>> (__istream_type & (*__pf)(__istream_type &))`
- `__istream_type & operator>> (__ios_type & (*__pf)(__ios_type &))`
- `__istream_type & operator>> (ios_base & (*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to false. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `__istream_type & operator>> (bool & __n)`
- `__istream_type & operator>> (short & __n)`
- `__istream_type & operator>> (unsigned short & __n)`
- `__istream_type & operator>> (int & __n)`
- `__istream_type & operator>> (unsigned int & __n)`
- `__istream_type & operator>> (long & __n)`
- `__istream_type & operator>> (unsigned long & __n)`
- `__istream_type & operator>> (long long & __n)`
- `__istream_type & operator>> (unsigned long long & __n)`
- `__istream_type & operator>> (float & __f)`
- `__istream_type & operator>> (double & __f)`
- `__istream_type & operator>> (long double & __f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to true. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type get ()`
 - `__istream_type & get (char_type &__c)`
 - `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
 - `__istream_type & get (char_type *__s, streamsize __n)`
 - `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
 - `__istream_type & get (__streambuf_type &__sb)`
 - `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `__istream_type & getline (char_type *__s, streamsize __n)`
 - `__istream_type & ignore (streamsize __n, int_type __delim)`
 - `__istream_type & ignore (streamsize __n)`
 - `__istream_type & ignore ()`
 - `int_type peek ()`
 - `__istream_type & read (char_type *__s, streamsize __n)`
 - `streamsize readsome (char_type *__s, streamsize __n)`
 - `__istream_type & putback (char_type __c)`
 - `__istream_type & unget ()`
 - `int sync ()`
 - `pos_type tellg ()`
 - `__istream_type & seekg (pos_type)`
 - `__istream_type & seekg (off_type, ios_base::seekdir)`
-
- `operator bool () const`
 - `bool operator! () const`

Static Public Member Functions

- `static bool sync_with_stdio (bool __sync=true)`
- `static int xalloc () throw ()`

Static Public Attributes

- `static const openmode __noreplace`
- `static const fmtflags adjustfield`
- `static const openmode app`
- `static const openmode ate`
- `static const iostate badbit`
- `static const fmtflags basefield`
- `static const seekdir beg`
- `static const openmode binary`
- `static const fmtflags boolalpha`
- `static const seekdir cur`
- `static const fmtflags dec`
- `static const seekdir end`
- `static const iostate eofbit`
- `static const iostate failbit`
- `static const fmtflags fixed`
- `static const fmtflags floatfield`
- `static const iostate goodbit`
- `static const fmtflags hex`
- `static const openmode in`
- `static const fmtflags internal`
- `static const fmtflags left`
- `static const fmtflags oct`
- `static const openmode out`

- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) & __loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename [_ValueT](#) >
 [_istream_type](#) & [_M_extract](#) ([_ValueT](#) & __v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) & __rhs) noexcept
- void [init](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > * __sb)
- void [move](#) ([basic_ios](#) && __rhs)
- void [move](#) ([basic_ios](#) & __rhs)
- void [set_rdbuf](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > * __sb)
- void [swap](#) ([basic_ios](#) & __rhs) noexcept
- void [swap](#) ([basic_istream](#) & __rhs)

Protected Attributes

- [_Callback_list](#) * [_M_callbacks](#)
- const [__ctype_type](#) * [_M_ctype](#)
- [iostate](#) [_M_exception](#)
- [char_type](#) [_M_fill](#)
- bool [_M_fill_init](#)
- [fmtflags](#) [_M_flags](#)
- [streamsize](#) [_M_gcount](#)
- [locale](#) [_M_ios_locale](#)
- [_Words](#) [_M_local_word](#) [[_S_local_word_size](#)]
- const [__num_get_type](#) * [_M_num_get](#)
- const [__num_put_type](#) * [_M_num_put](#)
- [streamsize](#) [_M_precision](#)
- [basic_streambuf](#)< [_CharT](#), [_Traits](#) > * [_M_streambuf](#)
- [iostate](#) [_M_streambuf_state](#)
- [basic_ostream](#)< [_CharT](#), [_Traits](#) > * [_M_tie](#)
- [streamsize](#) [_M_width](#)
- [_Words](#) * [_M_word](#)
- int [_M_word_size](#)
- [_Words](#) [_M_word_zero](#)

6.252.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_istream< _CharT, _Traits, _Alloc >
```

Controlling input for std::string.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports reading from objects of type `std::basic_string`, using the inherited functions from `std::basic_istream`. To control the associated sequence, an instance of `std::basic_stringbuf` is used, which this page refers to as `sb`.

6.252.2 Member Typedef Documentation

`__num_put_type`

```
template<typename _CharT , typename _Traits >
typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits
>::__num_put_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

`fmtflags`

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`

- internal
- left
- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- badbit
- eofbit
- failbit
- goodbit

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- app
- ate
- binary
- in
- out
- trunc

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.252.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.252.4 Constructor & Destructor Documentation**basic_istream() [1/3]**

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
std::basic_istream< _CharT, _Traits, _Alloc >::basic_istream ( ) [inline]
```

Default constructor starts with an empty string buffer.

Initializes `sb` using `in`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_istream() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
std::basic_istream< _CharT, _Traits, _Alloc >::basic_istream (
    ios_base::openmode __mode ) [inline], [explicit]
```

Starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

`ios_base::in` is automatically included in `__mode`.

Initializes `sb` using `__mode|in`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_istream() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
std::basic_istream< _CharT, _Traits, _Alloc >::basic_istream (
    const __string_type & __str,
    ios_base::openmode __mode = ios_base::in ) [inline], [explicit]
```

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
--------------------	--

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

`ios_base::in` is automatically included in *mode*.

Initializes *sb* using *str* and *mode*`|in`, and passes `&sb` to the base class initializer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

~basic_istream()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_istream< _CharT, _Traits, _Alloc >::~~basic_istream ( ) [inline]
```

The destructor does nothing.

The buffer is deallocated by the `stringbuf` object, not the formatting stream.

6.252.5 Member Function Documentation**_M_getloc()**

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_date()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::time_put< _CharT, _OutIter >::do_put()`, and `std::time_put< _CharT, _OutIter >::put()`.

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other `iostate` flags may also be set.

References `std::ios_base::badbit`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`.

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Referenced by `std::basic_ios<_CharT, _Traits>::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream<_CharT, _Traits>::putb`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<_CharT, _Traits>::sets` and `std::basic_istream<_CharT, _Traits>::unset()`.

copyfmt()

```
template<typename _CharT, typename _Traits>
basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt (
    const basic_ios<_CharT, _Traits> & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits>::exceptions()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT, typename _Traits>
bool std::basic_ios<_CharT, _Traits>::eof ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits>::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT, typename _Traits>
iostate std::basic_ios<_CharT, _Traits>::exceptions ( ) const [inline], [inherited]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline], [inherited]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::gcount ( ) const [inline], [inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::int_type std::basic_istream< _CharT, _Traits >::get (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline], [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim ) [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#), [std::basic_streambuf< _CharT, _Traits >::snextc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::setstate\(\)](#).

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

get() [6/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
------------------	----------------------

Parameters

<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

ignore() [1/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sbumpc\(\)](#).

ignore() [2/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

ignore() [3/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n,
    int_type __delim ) [inherited]
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A “stop” character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), and [std::ios_base::goodbit](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios< _CharT, _Traits >::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c,dfault)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

operator bool()

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator"!()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator>>() [1/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    __ios_type &(*)(__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.

For more information, see the `io manip` header.

operator>>() [2/17]

```
template<typename _CharT , typename _Traits >
```

```
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    __istream_type &(*)(__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`.

For more information, see the `io manip` header.

operator>>() [3/17]

```
template<typename _CharT , typename _Traits >
```

```
basic_istream< _CharT, _Traits > & std::basic_istream<_CharT, _Traits>::operator>> (
    __streambuf_type * __sb ) [inherited]
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, `failbit` is set.

References [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

operator>>() [4/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    bool & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

<code>f</code>	A variable of builtin floating point type.
----------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits >::operator>> (
    float & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_istream<_CharT, _Traits > & std::basic_istream<_CharT, _Traits >::operator>> (
    int & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

↵	A variable of builtin integral type.
<i>n</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream<_CharT, _Traits >::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as std::ws and std::dec use these functions in constructs like std::cin >> std::ws. For more information, see the iomanip header.

operator>>() [9/17]

```
template<typename _CharT , typename _Traits >
```



```
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<div><div>↔</div><div><div>__n</div></div></div>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

<div>↔</div>	A variable of builtin floating point type.
<div><div>↔</div><div>↔</div></div>	
<div>↔</div>	
<div><div>↔</div><div><i>f</i></div></div>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<div><div>↔</div><div>__n</div></div>	A variable of builtin integral type.
---------------------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned int & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [14/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [15/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [16/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [17/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    void *& __p ) [inline], [inherited]
```

Basic arithmetic extractors.

Parameters

 __p	A variable of pointer type.
--	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

peek()

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::peek (
    void ) [inherited]
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

putback()

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::putback (
    char_type __c ) [inherited]
```

Unextracting a single character.

Parameters

<code>_↔ _c</code>	The character to push back into the input stream.
------------------------	---

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf() -> sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References [std::basic_istream<_CharT, _Traits>::_M_gcount](#), [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sputbackc\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__stringbuf_type * std::basic_istream<_CharT, _Traits, _Alloc>::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current `basic_stringbuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (
    basic_streambuf<_CharT, _Traits> * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by [std::basic_ios< _CharT, _Traits >::bad\(\)](#), [std::basic_ios< _CharT, _Traits >::eof\(\)](#), [std::basic_ios< _CharT, _Traits >::fail\(\)](#), [std::basic_ios< _CharT, _Traits >::good\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::seek\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::setstate\(\)](#), and [std::basic_istream< _CharT, _Traits >::tellg\(\)](#).

read()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#).

readsome()

```
template<typename _CharT, typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::readsome (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf() -> in_avail()`, called A here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::min\(\)](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios< _CharT, _Traits >::fail](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (
    pos_type __pos ) [inherited]
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios< _CharT, _Traits >::fail](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set. Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

str() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__string_type std::basic_istream< _CharT, _Traits, _Alloc >::str ( ) const [inline]
```

Copying out the string buffer.

Returns

```
rdbuf ()->str ()
```

str() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_istream< _CharT, _Traits, _Alloc >::str (
    const __string_type & __s ) [inline]
```

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf ()->str (s)`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void ) [inherited]
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf ()` is a null pointer, returns -1.

Otherwise, calls `rdbuf ()->pubsync ()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount ()`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_streambuf< _CharT, _Traits >::pubsync\(\)](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void ) [inherited]
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::in`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tie() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::unget (
    void ) [inherited]
```

Unextracting the previous character.

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_streambuf<_CharT, _Traits>::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios<_CharT, _Traits>::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream<char, _Traits>::put()`.

`width()` [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

`width()` [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

`xalloc()`

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.252.6 Member Data Documentation

`_M_gcount`

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic_istream< _CharT, _Traits >::M_gcount [protected], [inherited]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::get()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< char >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_istream< _CharT, _Traits >::ungget()`, and `std::ws()`.

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, and `std::oct()`.

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```

Request a seek relative to the beginning of the stream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<char>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::read\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), and [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

fixed

const [fmtflags](#) std::ios_base::fixed [static], [inherited]

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

const [fmtflags](#) std::ios_base::floatfield [static], [inherited]

A mask of scientific|fixed. Useful for the 2-arg form of [setf](#).

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

const [iostate](#) std::ios_base::goodbit [static], [inherited]

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<_CharT, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unsetg\(\)](#), and [std::ws\(\)](#).

hex

const [fmtflags](#) std::ios_base::hex [static], [inherited]

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::hex\(\)](#).

in

const [openmode](#) std::ios_base::in [static], [inherited]

Open for input. Default for [ifstream](#) and [fstream](#).

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_filebuf<_CharT, _Traits>::underflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::xsgetn\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by [std::num_put< _CharT, _Outlter >::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put< _CharT, _Outlter >::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

const [fmtflags](#) `std::ios_base::skipws` [static], [inherited]

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

const [openmode](#) `std::ios_base::trunc` [static], [inherited]

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

const [fmtflags](#) `std::ios_base::unitbuf` [static], [inherited]

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

const [fmtflags](#) `std::ios_base::uppercase` [static], [inherited]

Replaces certain lowercase letters with their uppercase equivalents in generated output.

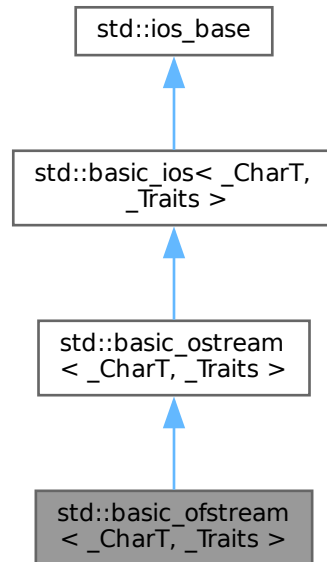
Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

6.253 std::basic_ofstream< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::basic_ofstream< _CharT, _Traits >:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_filebuf](#)< char_type, traits_type > **__filebuf_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
 - typedef [basic_ostream](#)< char_type, traits_type > **__ostream_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) (event __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags [fmtflags](#)
 - typedef traits_type::int_type **int_type**
 - typedef _ios_istate [iostate](#)
 - typedef traits_type::off_type **off_type**
 - typedef _ios_Openmode [openmode](#)
 - typedef traits_type::pos_type **pos_type**
 - typedef _ios_Seekdir [seekdir](#)
 - typedef _Traits **traits_type**
-
- typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**

Public Member Functions

- [basic_ofstream](#) ()
- **basic_ofstream** ([basic_ofstream](#) && __rhs)
- [template](#)<typename _Path , typename _Require = [_If_fs_path](#)<_Path>>
[basic_ofstream](#) (const _Path & __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- **basic_ofstream** (const [basic_ofstream](#) &)=delete
- [basic_ofstream](#) (const char * __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- [basic_ofstream](#) (const [std::string](#) & __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- [~basic_ofstream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- [template](#)<typename _ValueT >
[basic_ofstream](#)<_CharT, _Traits > & [_M_insert](#) (_ValueT __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- void [close](#) ()
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) & __rhs)
- bool [eof](#) () const
- [iostate exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- char_type [fill](#) () const
- char_type [fill](#) (char_type __ch)
- [fmtflags flags](#) () const
- [fmtflags flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [locale getloc](#) () const
- bool [good](#) () const
- [locale imbue](#) (const [locale](#) & __loc)
- bool [is_open](#) ()
- bool [is_open](#) () const
- long & [iword](#) (int __ix)
- char [narrow](#) (char_type __c, char __dfault) const
- [template](#)<typename _Path >
[_If_fs_path](#)<_Path, void > [open](#) (const _Path & __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- void [open](#) (const char * __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- void [open](#) (const [std::string](#) & __s, [ios_base::openmode](#) __mode=[ios_base::out](#))
- [__ostream_type](#) & [operator<<](#) ([__streambuf_type](#) * __sb)
- [__ostream_type](#) & [operator<<](#) (const void * __p)
- [__ostream_type](#) & [operator<<](#) (nullptr_t)
- [basic_ofstream](#) & [operator=](#) ([basic_ofstream](#) && __rhs)
- [basic_ofstream](#) & [operator=](#) (const [basic_ofstream](#) &)=delete
- [streamsize precision](#) () const
- [streamsize precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)
- [__filebuf_type](#) * [rdbuf](#) () const
- [basic_streambuf](#)<_CharT, _Traits > * [rdbuf](#) ([basic_streambuf](#)<_CharT, _Traits > * __sb)
- [iostate rdstate](#) () const
- void [register_callback](#) ([event_callback](#) __fn, int __index)
- [__ostream_type](#) & [seekp](#) (off_type, [ios_base::seekdir](#))

- `__ostream_type & seekp` (pos_type)
- `fmtflags setf` (fmtflags __fmtfl)
- `fmtflags setf` (fmtflags __fmtfl, fmtflags __mask)
- void `setstate` (iostate __state)
- void `swap` (basic_ofstream &__rhs)
- pos_type `tellp` ()
- `basic_ostream< _CharT, _Traits > * tie` () const
- `basic_ostream< _CharT, _Traits > * tie` (basic_ostream< _CharT, _Traits > *__tiestr)
- void `unsetf` (fmtflags __mask)
- char_type `widen` (char __c) const
- `streamsize width` () const
- `streamsize width` (streamsize __wide)
- `__ostream_type & operator<<` (__ostream_type &(*__pf)(__ostream_type &))
- `__ostream_type & operator<<` (__ios_type &(*__pf)(__ios_type &))
- `__ostream_type & operator<<` (ios_base &(*__pf)(ios_base &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<<` (long __n)
- `__ostream_type & operator<<` (unsigned long __n)
- `__ostream_type & operator<<` (bool __n)
- `__ostream_type & operator<<` (short __n)
- `__ostream_type & operator<<` (unsigned short __n)
- `__ostream_type & operator<<` (int __n)
- `__ostream_type & operator<<` (unsigned int __n)
- `__ostream_type & operator<<` (long long __n)
- `__ostream_type & operator<<` (unsigned long long __n)
- `__ostream_type & operator<<` (double __f)
- `__ostream_type & operator<<` (float __f)
- `__ostream_type & operator<<` (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put` (char_type __c)
- `__ostream_type & write` (const char_type *__s, streamsize __n)
- `operator bool` () const
- `bool operator!` () const

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [openmode](#) __noreplace
- static const [fmtflags](#) adjustfield
- static const [openmode](#) app
- static const [openmode](#) ate
- static const [iostate](#) badbit
- static const [fmtflags](#) basefield
- static const [seekdir](#) beg
- static const [openmode](#) binary
- static const [fmtflags](#) boolalpha
- static const [seekdir](#) cur
- static const [fmtflags](#) dec
- static const [seekdir](#) end
- static const [iostate](#) eofbit
- static const [iostate](#) failbit
- static const [fmtflags](#) fixed
- static const [fmtflags](#) floatfield
- static const [iostate](#) goodbit
- static const [fmtflags](#) hex
- static const [openmode](#) in
- static const [fmtflags](#) internal
- static const [fmtflags](#) left
- static const [fmtflags](#) oct
- static const [openmode](#) out
- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) & __loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename _ValueT >
[__ostream_type](#) & [_M_insert](#) (_ValueT __v)

- void **_M_move** (ios_base &) noexcept
- void **_M_swap** (ios_base &__rhs) noexcept
- void **init** (basic_streambuf<_CharT, _Traits> *__sb)
- void **move** (basic_ios &&__rhs)
- void **move** (basic_ios &__rhs)
- void **set_rdbuf** (basic_streambuf<_CharT, _Traits> *__sb)
- void **swap** (basic_ios &__rhs) noexcept
- void **swap** (basic_ostream &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const __ctype_type * **_M_ctype**
- iostate **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- fmtflags **_M_flags**
- locale **_M_ios_locale**
- _Words **_M_local_word** [_S_local_word_size]
- const __num_get_type * **_M_num_get**
- const __num_put_type * **_M_num_put**
- streamsize **_M_precision**
- basic_streambuf<_CharT, _Traits> * **_M_streambuf**
- iostate **_M_streambuf_state**
- basic_ostream<_CharT, _Traits> * **_M_tie**
- streamsize **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

6.253.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ofstream<_CharT, _Traits>
```

Controlling output for files.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class supports reading from named files, using the inherited functions from `std::basic_ostream`. To control the associated sequence, an instance of `std::basic_filebuf` is used, which this page refers to as `sb`.

6.253.2 Member Typedef Documentation

`__num_get_type`

```
template<typename _CharT, typename _Traits>
typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits> > std::basic_ios<_CharT, _Traits>
::__num_get_type [inherited]
```

These are non-standard types.

event_callback

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>↵ __e</code>	One of the members of the event enum.
<code>↵ __b</code>	Reference to the ios_base object.
<code>↵ __i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- boolalpha
- dec
- fixed
- hex
- internal
- left
- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.253.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.253.4 Constructor & Destructor Documentation

basic_ofstream() [1/4]

```
template<typename _CharT , typename _Traits >
std::basic_ofstream< _CharT, _Traits >::basic_ofstream ( ) [inline]
```

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

basic_ofstream() [2/4]

```
template<typename _CharT , typename _Traits >
std::basic_ofstream< _CharT, _Traits >::basic_ofstream (
    const char * __s,
    ios_base::openmode __mode = ios_base::out ) [inline], [explicit]
```

Create an output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

`ios_base::out` is automatically included in `__mode`.

basic_ofstream() [3/4]

```
template<typename _CharT , typename _Traits >
std::basic_ofstream< _CharT, _Traits >::basic_ofstream (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::out ) [inline], [explicit]
```

Create an output file stream.

Parameters

<code>__s</code>	<code>std::string</code> specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

`ios_base::out` is automatically included in `__mode`.

basic_ofstream() [4/4]

```
template<typename _CharT , typename _Traits >
template<typename _Path , typename _Require = _If_fs_path<_Path>>
std::basic_ofstream< _CharT, _Traits >::basic_ofstream (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::out ) [inline]
```

Create an output file stream.

Parameters

<code>__s</code>	<code>filesystem::path</code> specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

`ios_base::out` is automatically included in `__mode`.

~basic_ofstream()

```
template<typename _CharT , typename _Traits >
std::basic_ofstream< _CharT, _Traits >::~~basic_ofstream ( ) [inline]
```

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

6.253.5 Member Function Documentation

_M_getloc()

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by `std::money_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_put< _CharT, _Outlter >::do_put()`, and `std::time_put< _CharT, _Outlter >::put()`.

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References `std::ios_base::badbit`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`.

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Referenced by `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putb()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< _CharT, _Traits >::setsb()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

close()

```
template<typename _CharT , typename _Traits >
void std::basic_ofstream< _CharT, _Traits >::close ( ) [inline]
```

Close the file.

Calls `std::basic_filebuf::close()`. If that function fails, `failbit` is set in the stream's error state.

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (
    const basic_ios< _CharT, _Traits > & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits>::exceptions()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::eof ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the `eofbit` is set.

Note that other `iosstate` flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits>::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::exceptions ( ) const [inline], [inherited]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iosstate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline], [inherited]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other `iostate` flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ofstream< _CharT, _Traits > & std::basic_ofstream< _CharT, _Traits >::flush [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf() ->pubsync()`, and if that returns -1, sets `badbit`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits`

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, `std::operator>>()`, and `std::ws()`.

good()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::__detail::operator>>()`.

imbue()

```
template<typename _CharT , typename _Traits >
```

```
locale std::basic_ios<_CharT, _Traits>::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by [std::basic_ios< _CharT, _Traits >::basic_ios\(\)](#).

is_open()

```
template<typename _CharT , typename _Traits >
bool std::basic_ofstream< _CharT, _Traits >::is_open ( ) [inline]
```

Wrapper to test for an open file.

Returns


```
rdbuf() -> is_open()
```

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

 <code>__ix</code>	Index into the array.
---	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>>>(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

open() [1/3]

```
template<typename _CharT , typename _Traits >
template<typename _Path >
_If_fs_path< _Path, void > std::basic_ofstream< _CharT, _Traits >::open (
    const _Path & __s,
    ios_base::openmode __mode = ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a <code>filesystem::path</code> .
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s,__mode|out)`. If that function fails, `failbit` is set in the stream's error state.

open() [2/3]

```
template<typename _CharT , typename _Traits >
void std::basic_ofstream< _CharT, _Traits >::open (
    const char * __s,
    ios_base::openmode __mode = ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s,__mode|out)`. If that function fails, `failbit` is set in the stream's error state.

open() [3/3]

```
template<typename _CharT , typename _Traits >
void std::basic_ofstream< _CharT, _Traits >::open (
    const std::string & __s,
    ios_base::openmode __mode = ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(s,mode|out)`. If that function fails, `failbit` is set in the stream's error state.

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator"!()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ostream_type &(*) (__ostream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [3/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    __streambuf_type * __sb ) [inherited]
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set `failbit` in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted),

or

- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

operator<<() [4/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [5/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    const void * __p ) [inline], [inherited]
```

Pointer arithmetic inserters.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

<code>f</code>	A variable of builtin floating point type.
----------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ofstream< _CharT, _Traits >::operator<< (
    float __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
basic_ofstream< _CharT, _Traits > & std::basic_ofstream< _CharT, _Traits >::operator<< (
    int __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

↵	A variable of builtin integral type.
<i>n</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ofstream< _CharT, _Traits >::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as std::endl and std::hex use these functions in constructs like "std::cout << std::endl". For more information, see the iomanip header.

operator<<() [10/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

\leftarrow __ \leftarrow \leftarrow __ \leftarrow f	A variable of builtin floating point type.
---	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

\leftarrow __n	A variable of builtin integral type.
---------------------	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [13/17]

```
template<typename _CharT , typename _Traits >
basic_ofstream< _CharT, _Traits > & std::basic_ofstream< _CharT, _Traits >::operator<< (
    short __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.


References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::num_put< _CharT, _Outiter >::put\(\)](#), and [std::basic_ios< _CharT, _Traits >](#)

operator<<() [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ofstream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [15/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ofstream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 __n	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned short __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

put()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (
    char_type __c ) [inherited]
```

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

*this

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
__filebuf_type * std::basic_ofstream< _CharT, _Traits >::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current basic_filebuf buffer.

This hides both signatures of std::basic_ios::rdbuf().

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_streambuf< _CharT, _Traits > * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See std::ios_base::iostate for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekf()`, `std::basic_istream< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::tellg()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekp() [1/2]

```
template<typename _CharT, typename _Traits>
basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

seekp() [2/2]

```
template<typename _CharT, typename _Traits>
basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (
    pos_type __pos ) [inherited]
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set. Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellp()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp [inherited]
Getting the current write position.
```

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`

tie() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Referenced by `std::basic_ios< _CharT, _Traits >::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ofstream< char, _Traits >::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

write()

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (
    const char_type * __s,
    streamsize __n ) [inherited]
```

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.253.6 Member Data Documentation

adjustfield

const `fmtflags` std::ios_base::adjustfield [static], [inherited]

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

const `openmode` std::ios_base::app [static], [inherited]

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

const `openmode` std::ios_base::ate [static], [inherited]

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

badbit

const `istate` std::ios_base::badbit [static], [inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::get()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< char >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_istream< _CharT, _Traits >::ung` and `std::ws()`.

basefield

const `fmtflags` std::ios_base::basefield [static], [inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get< _CharT, _Initer >::do_get()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::hex()`, and `std::oct()`.

beg

const `seekdir` std::ios_base::beg [static], [inherited]

Request a seek relative to the beginning of the stream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`.

binary

const `openmode` std::ios_base::binary [static], [inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by `std::basic_filebuf< _CharT, _Traits >::showmanyc()`.

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::time_get< _CharT, _Inlter >::do_get\(\)](#),

[std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_year\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#),

[std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::time_get< _CharT, _Inlter >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#),

[std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#),

[std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsomewhat\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::num_put< _CharT, _Outlter >::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_filebuf< _CharT, _Traits >::pbackfail\(\)](#),

[std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos\(\)](#), [std::basic_filebuf< _CharT, _Traits >::showmanyc\(\)](#),

[std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_filebuf< _CharT, _Traits >::underflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::xsgetn\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.) Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::pbackfail\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::tellp\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

`const openmode std::ios_base::trunc` [static], [inherited]
Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

const `fmt::ios_base::unitbuf` [static], [inherited]
 Flushes output after each output operation.
 Referenced by `std::nounitbuf()`, and `std::unitbuf()`.

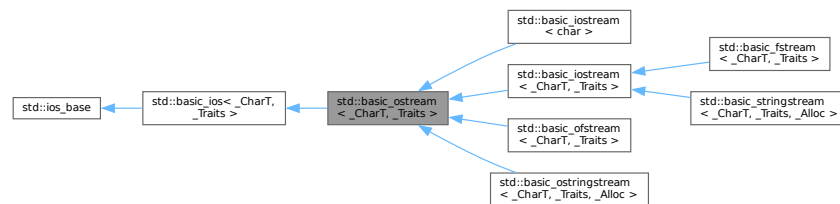
uppercase

const `fmt::ios_base::uppercase` [static], [inherited]
 Replaces certain lowercase letters with their uppercase equivalents in generated output.
 Referenced by `std::num_put<_CharT, _OutIter>::do_put()`, `std::nouppercase()`, and `std::uppercase()`.
 The documentation for this class was generated from the following file:

- fstream

6.254 `std::basic_ostream<_CharT, _Traits>` Class Template Reference

Inheritance diagram for `std::basic_ostream<_CharT, _Traits>`:



Classes

- class sentry

Public Types

- typedef `ctype` < _CharT > **__ctype_type**
- typedef `basic_ios` < _CharT, _Traits > **__ios_type**
- typedef `num_put` < _CharT, `ostreambuf_iterator` < _CharT, _Traits > > **__num_put_type**
- typedef `basic_ostream` < _CharT, _Traits > **__ostream_type**
- typedef `basic_streambuf` < _CharT, _Traits > **__streambuf_type**
- typedef _CharT **char_type**
- enum `event` { `erase_event` , `imbue_event` , `copyfmt_event` }
- typedef void(* `event_callback`) (`event` __e, `ios_base` &__b, int __i)
- typedef _ios_Fmtflags **fmtflags**
- typedef _Traits::int_type **int_type**
- typedef _ios_istate **istate**
- typedef _Traits::off_type **off_type**
- typedef _ios_Openmode **openmode**

- typedef `_Traits::pos_type` **pos_type**
 - typedef `_ios_Seekdir` [seekdir](#)
 - typedef `_Traits` **traits_type**
-
- typedef `num_get<_CharT, istreambuf_iterator<_CharT, _Traits>>` [__num_get_type](#)

Public Member Functions

- [basic_ostream](#) ([__streambuf_type](#) *__sb)
- virtual [~basic_ostream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- template<typename `_ValueT` >
[basic_ostream](#)<`_CharT`, `_Traits` > & [_M_insert](#) (`_ValueT` __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- `char_type` [fill](#) () const
- `char_type` [fill](#) (`char_type` __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [locale](#) [getloc](#) () const
- bool [good](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc)
- long & [iword](#) (int __ix)
- `char` [narrow](#) (`char_type` __c, `char` __dfault) const
- [__ostream_type](#) & [operator<<](#) ([__streambuf_type](#) *__sb)
- [__ostream_type](#) & [operator<<](#) (const void *__p)
- [__ostream_type](#) & [operator<<](#) (nullptr_t)
- [streamsize](#) [precision](#) () const
- [streamsize](#) [precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)
- [basic_streambuf](#)<`_CharT`, `_Traits` > * [rdbuf](#) () const
- [basic_streambuf](#)<`_CharT`, `_Traits` > * [rdbuf](#) ([basic_streambuf](#)<`_CharT`, `_Traits` > *__sb)
- [iostate](#) [rdstate](#) () const
- void [register_callback](#) ([event_callback](#) __fn, int __index)
- [__ostream_type](#) & [seekp](#) (`off_type`, [ios_base::seekdir](#))
- [__ostream_type](#) & [seekp](#) (`pos_type`)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
- void [setstate](#) ([iostate](#) __state)
- `pos_type` [tellp](#) ()
- [basic_ostream](#)<`_CharT`, `_Traits` > * [tie](#) () const
- [basic_ostream](#)<`_CharT`, `_Traits` > * [tie](#) ([basic_ostream](#)<`_CharT`, `_Traits` > *__tiestr)
- void [unsetf](#) ([fmtflags](#) __mask)
- `char_type` [widen](#) (`char` __c) const

- [streamsize width](#) () const
- [streamsize width](#) (streamsize __wide)
- [__ostream_type & operator<<](#) (__ostream_type &(*__pf)(__ostream_type &))
- [__ostream_type & operator<<](#) (__ios_type &(*__pf)(__ios_type &))
- [__ostream_type & operator<<](#) (ios_base &(*__pf)(ios_base &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- [__ostream_type & operator<<](#) (long __n)
- [__ostream_type & operator<<](#) (unsigned long __n)
- [__ostream_type & operator<<](#) (bool __n)
- [__ostream_type & operator<<](#) (short __n)
- [__ostream_type & operator<<](#) (unsigned short __n)
- [__ostream_type & operator<<](#) (int __n)
- [__ostream_type & operator<<](#) (unsigned int __n)
- [__ostream_type & operator<<](#) (long long __n)
- [__ostream_type & operator<<](#) (unsigned long long __n)
- [__ostream_type & operator<<](#) (double __f)
- [__ostream_type & operator<<](#) (float __f)
- [__ostream_type & operator<<](#) (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- [__ostream_type & put](#) (char_type __c)
- [__ostream_type & write](#) (const char_type *__s, streamsize __n)
- [operator bool](#) () const
- [bool operator!](#) () const

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [openmode](#) `__noreplace`
- static const [fmtflags](#) `adjustfield`
- static const [openmode](#) `app`
- static const [openmode](#) `ate`
- static const [iostate](#) `badbit`
- static const [fmtflags](#) `basefield`
- static const [seekdir](#) `beg`
- static const [openmode](#) `binary`
- static const [fmtflags](#) `boolalpha`
- static const [seekdir](#) `cur`
- static const [fmtflags](#) `dec`
- static const [seekdir](#) `end`
- static const [iostate](#) `eofbit`
- static const [iostate](#) `failbit`
- static const [fmtflags](#) `fixed`
- static const [fmtflags](#) `floatfield`
- static const [iostate](#) `goodbit`
- static const [fmtflags](#) `hex`
- static const [openmode](#) `in`
- static const [fmtflags](#) `internal`
- static const [fmtflags](#) `left`
- static const [fmtflags](#) `oct`
- static const [openmode](#) `out`
- static const [fmtflags](#) `right`
- static const [fmtflags](#) `scientific`
- static const [fmtflags](#) `showbase`
- static const [fmtflags](#) `showpoint`
- static const [fmtflags](#) `showpos`
- static const [fmtflags](#) `skipws`
- static const [openmode](#) `trunc`
- static const [fmtflags](#) `unitbuf`
- static const [fmtflags](#) `uppercase`

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- **basic_ostream** ([basic_iostream](#)< [_CharT](#), [_Traits](#) > &)
- **basic_ostream** ([basic_ostream](#) && [__rhs](#))
- **basic_ostream** (const [basic_ostream](#) &)=delete
- void [_M_cache_locale](#) (const [locale](#) & [__loc](#))
- void [_M_call_callbacks](#) ([event](#) [__ev](#)) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- [_Words](#) & [_M_grow_words](#) (int [__index](#), bool [__iword](#))
- void [_M_init](#) () throw ()
- template<typename [_ValueT](#) >
 [__ostream_type](#) & [_M_insert](#) ([_ValueT](#) [__v](#))
- void [_M_move](#) ([ios_base](#) &) noexcept

- void **_M_swap** (ios_base &__rhs) noexcept
- void **init** (basic_streambuf<_CharT, _Traits> *__sb)
- void **move** (basic_ios &&__rhs)
- void **move** (basic_ios &__rhs)
- **basic_ostream** & **operator=** (basic_ostream &&__rhs)
- **basic_ostream** & **operator=** (const basic_ostream &)=delete
- void **set_rdbuf** (basic_streambuf<_CharT, _Traits> *__sb)
- void **swap** (basic_ios &__rhs) noexcept
- void **swap** (basic_ostream &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const __ctype_type * **_M_ctype**
- iostate **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- fmtflags **_M_flags**
- locale **_M_ios_locale**
- _Words **_M_local_word** [_S_local_word_size]
- const __num_get_type * **_M_num_get**
- const __num_put_type * **_M_num_put**
- streamsize **_M_precision**
- basic_streambuf<_CharT, _Traits> * **_M_streambuf**
- iostate **_M_streambuf_state**
- basic_ostream<_CharT, _Traits> * **_M_tie**
- streamsize **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

Friends

- class **sentry**

6.254.1 Detailed Description

template<typename _CharT, typename _Traits>

class std::basic_ostream<_CharT, _Traits>

Template class basic_ostream.

Template Parameters

<i>_CharT</i>	Type of character stream.
<i>_Traits</i>	Traits for character type, defaults to char_traits<_CharT>.

This is the base class for all output streams. It provides text formatting of all builtin types, and communicates with any class derived from basic_streambuf to do the actual output.

6.254.2 Member Typedef Documentation

`__num_get_type`

```
template<typename _CharT , typename _Traits >
typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits
>::__num_get_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the ios_base object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

`fmtflags`

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`

- uppercase
- adjustfield
- basefield
- floatfield

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- badbit
- eofbit
- failbit
- goodbit

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- app
- ate
- binary
- in
- out
- trunc

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- beg
- cur, equivalent to `SEEK_CUR` in the C standard library.
- end, equivalent to `SEEK_END` in the C standard library.

6.254.3 Member Enumeration Documentation

event

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.254.4 Constructor & Destructor Documentation

basic_ostream()

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits >::basic_ostream (
    __streambuf_type * __sb ) [inline], [explicit]
```

Base constructor.

This ctor is almost never called by the user directly, rather from derived classes' initialization lists, which pass a pointer to their own stream buffer.

~basic_ostream()

```
template<typename _CharT , typename _Traits >
virtual std::basic_ostream< _CharT, _Traits >::~~basic_ostream ( ) [inline], [virtual]
```

Base destructor.

This does very little apart from providing a virtual base dtor.

6.254.5 Member Function Documentation

_M_getloc()

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by `std::money_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_put< _CharT, _Outlter >::do_put()`, and `std::time_put< _CharT, _Outlter >::put()`.

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other `iostate` flags may also be set.

References `std::ios_base::badbit`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`.

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios<_CharT, _Traits>::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::putb](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ios<_CharT, _Traits>::sets](#) and [std::basic_istream<_CharT, _Traits>::unset\(\)](#).

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt (
    const basic_ios<_CharT, _Traits> & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

References [std::__addressof\(\)](#), [std::basic_ios<_CharT, _Traits>::exceptions\(\)](#), [std::basic_ios<_CharT, _Traits>::fill\(\)](#), [std::ios_base::flags\(\)](#), [std::ios_base::getloc\(\)](#), [std::ios_base::precision\(\)](#), [std::basic_ios<_CharT, _Traits>::tie\(\)](#), [std::tie\(\)](#), and [std::ios_base::width\(\)](#).

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::eof ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

References [std::ios_base::eofbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios<_CharT, _Traits>::exceptions ( ) const [inline], [inherited]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#).

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline], [inherited]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other `iostate` flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::flush
Synchronizing the stream buffer.
```

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>](#)

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, `std::operator>>()`, and `std::ws()`.

good()

```
template<typename _CharT, typename _Traits>
```

```
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::__detail::operator>>()`.

imbue()

```
template<typename _CharT, typename _Traits>
```

```
locale std::basic_ios<_CharT, _Traits>::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT, typename _Traits>
```

```
void std::basic_ios<_CharT, _Traits>::init (
    basic_streambuf<_CharT, _Traits> * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios<_CharT, _Traits>::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator"!()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#).

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    __ios_type &(*) (__ios_type &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    __ostream_type &(*) (__ostream_type &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [3/17]

```
template<typename _CharT , typename _Traits >
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<< (
    __streambuf_type * __sb )
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets `failbit` in the error state

If the function inserts no characters, `failbit` is set.

operator<<() [4/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    bool __n ) [inline]
```

Integer arithmetic inserters.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

$\ast this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [5/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    const void * __p ) [inline]
```

Pointer arithmetic inserters.

Parameters

$_ \leftarrow$	A variable of pointer type.
$_p$	

Returns

$\ast this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline]
```

Floating point arithmetic inserters.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
f	

Returns

$\ast this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    float __f ) [inline]
```

Floating point arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    int __n )
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `io manip` header.

operator<<() [10/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long __n ) [inline]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long double __f ) [inline]
```

Floating point arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_ n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [13/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    short __n )
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_ n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::num_put<_CharT, _OutIter>::put()`, and `std::basic_ios<_CharT, _Traits>`

operator<<() [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [15/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT, typename _Traits>
__ostream_type & std::basic_ostream<_CharT, _Traits>::operator<< (
    unsigned short __n ) [inline]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

put()

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::put (
    char_type __c )
```

Simple insertion.

Parameters

<code>_↔</code>	The character to insert.
<code>_C</code>	

Returns

*this

Tries to insert `__C`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>_↔</code>	Index into the array.
<code>_ix</code>	

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf ( ) const [inline],
[inherited]
```

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<char>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::tr2::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::peek\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<char>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

[std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_ostream<_CharT, _Traits>::tellp\(\)](#), [std::basic_istream<_CharT, _Traits>::unget\(\)](#), and [std::ws\(\)](#).

rdbuf() [2/2]

```
template<typename _CharT, typename _Traits>
basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (
    basic_streambuf<_CharT, _Traits> * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT, typename _Traits>
iostate std::basic_ios<_CharT, _Traits>::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by [std::basic_ios<_CharT, _Traits>::bad\(\)](#), [std::basic_ios<_CharT, _Traits>::eof\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_ios<_CharT, _Traits>::good\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::setstate\(\)](#), and [std::basic_istream<_CharT, _Traits>::tellg\(\)](#).

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekp() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (
    off_type __off,
    ios_base::seekdir __dir )
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (
    pos_type __pos )
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ostream< _CharT, _Traits >::sentry::sentry\(\)](#),

[std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::getline\(\)](#), [std::getline\(\)](#),

[std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT,](#)

[std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits](#)

[std::tr2::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#),

[std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits](#)

[std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >](#)

[std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellp()

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp
```

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`

tie() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Referenced by `std::basic_ios< _CharT, _Traits >::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream< char, _Traits >::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

write()

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (
    const char_type * __s,
    streamsize __n )
```

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( )    [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.254.6 Member Data Documentation

adjustfield

const `fmtflags` std::ios_base::adjustfield [static], [inherited]

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put<_CharT, _Outiter>::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

const `openmode` std::ios_base::app [static], [inherited]

Seek to end before each write.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

ate

const `openmode` std::ios_base::ate [static], [inherited]

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`.

badbit

const `iostate` std::ios_base::badbit [static], [inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ios<_CharT, _Traits>::bad()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::get()`, `std::basic_ostream<char, _Traits>::operator<<()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::basic_ostream<char, _Traits>::operator<<()`, `std::operator>>()`, `std::basic_istream<char>::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<char>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

basefield

const `fmtflags` std::ios_base::basefield [static], [inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get<_CharT, _Initer>::do_get()`, `std::num_put<_CharT, _Outiter>::do_put()`, `std::hex()`, and `std::oct()`.

beg

const `seekdir` std::ios_base::beg [static], [inherited]

Request a seek relative to the beginning of the stream.

Referenced by `std::basic_filebuf<_CharT, _Traits>::seekpos()`.

binary

const `openmode` std::ios_base::binary [static], [inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by `std::basic_filebuf<_CharT, _Traits>::showmanyc()`.

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::time_get< _CharT, _Inlter >::do_get\(\)](#),

[std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get< _CharT, _Inlter >::do_get_year\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#),

[std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::time_get< _CharT, _Inlter >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#),

[std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::operator>>\(\)](#),

[std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get< _CharT, _Inlter >::do_get\(\)](#), [std::num_put< _CharT, _Outlter >::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_filebuf< _CharT, _Traits >::pbackfail\(\)](#),

[std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos\(\)](#), [std::basic_filebuf< _CharT, _Traits >::showmanyc\(\)](#),

[std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_filebuf< _CharT, _Traits >::underflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::xsgetn\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.) Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::pbackfail\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::tellp\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for ofstream.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nunitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

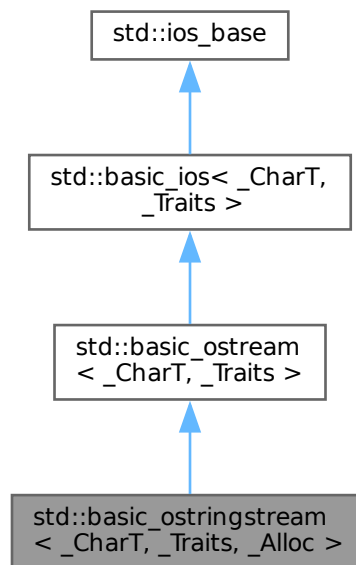
Referenced by [std::num_put< _CharT, _Outiter >::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [ostream](#)
- [ostream.tcc](#)

6.255 std::basic_ostringstream< _CharT, _Traits, _Alloc > Class Template Reference

Inheritance diagram for std::basic_ostringstream< _CharT, _Traits, _Alloc >:

**Public Types**

- typedef [ctype< _CharT >](#) `__ctype_type`

- typedef [basic_ios](#)< [_CharT](#), [_Traits](#) > [__ios_type](#)
 - typedef [num_put](#)< [_CharT](#), [ostreambuf_iterator](#)< [_CharT](#), [_Traits](#) > > [__num_put_type](#)
 - typedef [basic_ostream](#)< [char_type](#), [traits_type](#) > [__ostream_type](#)
 - typedef [basic_streambuf](#)< [_CharT](#), [_Traits](#) > [__streambuf_type](#)
 - typedef [basic_string](#)< [_CharT](#), [_Traits](#), [_Alloc](#) > [__string_type](#)
 - typedef [basic_stringbuf](#)< [_CharT](#), [_Traits](#), [_Alloc](#) > [__stringbuf_type](#)
 - typedef [_Alloc](#) [allocator_type](#)
 - typedef [_CharT](#) [char_type](#)
 - enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) & __b, int __i)
 - typedef [_ios_Fmtflags](#) [fmtflags](#)
 - typedef [traits_type::int_type](#) [int_type](#)
 - typedef [_ios_iostate](#) [iostate](#)
 - typedef [traits_type::off_type](#) [off_type](#)
 - typedef [_ios_Openmode](#) [openmode](#)
 - typedef [traits_type::pos_type](#) [pos_type](#)
 - typedef [_ios_Seekdir](#) [seekdir](#)
 - typedef [_Traits](#) [traits_type](#)
-
- typedef [num_get](#)< [_CharT](#), [istreambuf_iterator](#)< [_CharT](#), [_Traits](#) > > [__num_get_type](#)

Public Member Functions

- [basic_ostringstream](#) ()
- [basic_ostringstream](#) ([basic_ostringstream](#) && __rhs)
- [basic_ostringstream](#) (const [__string_type](#) & __str, [ios_base::openmode](#) __mode=[ios_base::out](#))
- [basic_ostringstream](#) (const [basic_ostringstream](#) &)=delete
- [basic_ostringstream](#) ([ios_base::openmode](#) __mode)
- [~basic_ostringstream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- template<typename [_ValueT](#) >
[basic_ostream](#)< [_CharT](#), [_Traits](#) > & [_M_insert](#) ([_ValueT](#) __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) & __rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- [char_type](#) [fill](#) () const
- [char_type](#) [fill](#) ([char_type](#) __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [locale](#) [getloc](#) () const
- bool [good](#) () const
- [locale](#) [imbue](#) (const [locale](#) & __loc)
- long & [iword](#) (int __ix)
- [char](#) [narrow](#) ([char_type](#) __c, [char](#) __dfault) const
- [__ostream_type](#) & [operator<<](#) ([__streambuf_type](#) * __sb)

- [__ostream_type](#) & [operator<<](#) (const void * __p)
 - [__ostream_type](#) & [operator<<](#) (nullptr_t)
 - [basic_ostringstream](#) & [operator=](#) ([basic_ostringstream](#) && __rhs)
 - [basic_ostringstream](#) & [operator=](#) (const [basic_ostringstream](#) &)=delete
 - [streamsize](#) [precision](#) () const
 - [streamsize](#) [precision](#) ([streamsize](#) __prec)
 - void *& [pword](#) (int __ix)
 - [__stringbuf_type](#) * [rdbuf](#) () const
 - [basic_streambuf](#)<_CharT, _Traits> * [rdbuf](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
 - [iostate](#) [rdstate](#) () const
 - void [register_callback](#) ([event_callback](#) __fn, int __index)
 - [__ostream_type](#) & [seekp](#) (off_type, [ios_base::seekdir](#))
 - [__ostream_type](#) & [seekp](#) (pos_type)
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
 - void [setstate](#) ([iostate](#) __state)
 - [__string_type](#) [str](#) () const
 - void [str](#) (const [__string_type](#) & __s)
 - void [swap](#) ([basic_ostringstream](#) & __rhs)
 - pos_type [tellp](#) ()
 - [basic_ostream](#)<_CharT, _Traits> * [tie](#) () const
 - [basic_ostream](#)<_CharT, _Traits> * [tie](#) ([basic_ostream](#)<_CharT, _Traits> * __tiestr)
 - void [unsetf](#) ([fmtflags](#) __mask)
 - char_type [widen](#) (char __c) const
 - [streamsize](#) [width](#) () const
 - [streamsize](#) [width](#) ([streamsize](#) __wide)
-
- [__ostream_type](#) & [operator<<](#) ([__ostream_type](#) &(* __pf)([__ostream_type](#) &))
 - [__ostream_type](#) & [operator<<](#) ([__ios_type](#) &(* __pf)([__ios_type](#) &))
 - [__ostream_type](#) & [operator<<](#) ([ios_base](#) &(* __pf)([ios_base](#) &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- [__ostream_type](#) & [operator<<](#) (long __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned long __n)
 - [__ostream_type](#) & [operator<<](#) (bool __n)
 - [__ostream_type](#) & [operator<<](#) (short __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned short __n)
 - [__ostream_type](#) & [operator<<](#) (int __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned int __n)
 - [__ostream_type](#) & [operator<<](#) (long long __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned long long __n)
-
- [__ostream_type](#) & [operator<<](#) (double __f)
 - [__ostream_type](#) & [operator<<](#) (float __f)

- `__ostream_type & operator<<` (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If badbit is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put` (char_type __c)
- `__ostream_type & write` (const char_type *__s, streamsize __n)
- `operator bool` () const
- `bool operator!` () const

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const `openmode` `__noreplace`
- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`
- static const `fmtflags` `basefield`
- static const `seekdir` `beg`
- static const `openmode` `binary`
- static const `fmtflags` `boolalpha`
- static const `seekdir` `cur`
- static const `fmtflags` `dec`
- static const `seekdir` `end`
- static const `iosstate` `eofbit`
- static const `iosstate` `failbit`
- static const `fmtflags` `fixed`
- static const `fmtflags` `floatfield`
- static const `iosstate` `goodbit`
- static const `fmtflags` `hex`
- static const `openmode` `in`
- static const `fmtflags` `internal`
- static const `fmtflags` `left`
- static const `fmtflags` `oct`
- static const `openmode` `out`
- static const `fmtflags` `right`
- static const `fmtflags` `scientific`
- static const `fmtflags` `showbase`
- static const `fmtflags` `showpoint`
- static const `fmtflags` `showpos`
- static const `fmtflags` `skipws`
- static const `openmode` `trunc`
- static const `fmtflags` `unitbuf`
- static const `fmtflags` `uppercase`

Protected Types

- enum { **_S_local_word_size** }

Protected Member Functions

- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- _Words & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- template<typename _ValueT>
[__ostream_type](#) & **_M_insert** (_ValueT __v)
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)<_CharT, _Traits> *__sb)
- void **move** ([basic_ios](#) &&__rhs)
- void **move** ([basic_ios](#) &__rhs)
- void **set_rdbuf** ([basic_streambuf](#)<_CharT, _Traits> *__sb)
- void **swap** ([basic_ios](#) &__rhs) noexcept
- void **swap** ([basic_ostream](#) &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [locale](#) **_M_ios_locale**
- _Words **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)<_CharT, _Traits> * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)<_CharT, _Traits> * **_M_tie**
- [streamsize](#) **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

6.255.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_ostringstream<_CharT, _Traits, _Alloc>
```

Controlling output for std::string.

Template Parameters

<i>_CharT</i>	Type of character stream.
<i>_Traits</i>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<i>_Alloc</i>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports writing to objects of type `std::basic_string`, using the inherited functions from `std::basic_ostream`. To control the associated sequence, an instance of `std::basic_stringbuf` is used, which this page refers to as `sb`.

6.255.2 Member Typedef Documentation

`__num_get_type`

```
template<typename _CharT, typename _Traits>
typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits> > std::basic_ios<_CharT, _Traits>::__num_get_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

`fmtflags`

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`

- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

`iosstate`

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

`openmode`

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

`seekdir`

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.255.3 Member Enumeration Documentation

event

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.255.4 Constructor & Destructor Documentation

basic_ostringstream() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_ostringstream< _CharT, _Traits, _Alloc >::basic_ostringstream ( ) [inline]
```

Default constructor starts with an empty string buffer.

Initializes `sb` using `mode|out`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_ostringstream() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_ostringstream< _CharT, _Traits, _Alloc >::basic_ostringstream (
    ios_base::openmode __mode ) [inline], [explicit]
```

Starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

`ios_base::out` is automatically included in `mode`.

Initializes `sb` using `mode|out`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_ostringstream() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_ostringstream< _CharT, _Traits, _Alloc >::basic_ostringstream (
    const __string_type & __str,
    ios_base::openmode __mode = ios_base::out ) [inline], [explicit]
```

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__mode</code>	Whether the buffer can read, or write, or both.

`ios_base::out` is automatically included in `mode`.

Initializes `sb` using `str` and `mode|out`, and passes `&sb` to the base class initializer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

~basic_ostringstream()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_ostringstream< _CharT, _Traits, _Alloc >::~~basic_ostringstream ( ) [inline]
```

The destructor does nothing.

The buffer is deallocated by the stringbuf object, not the formatting stream.

6.255.5 Member Function Documentation

_M_getloc()

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Referenced by [std::money_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_date\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_weekday\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_year\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), [std::time_put<_CharT, _Outlter>::do_put\(\)](#), and [std::time_put<_CharT, _Outlter>::put\(\)](#).

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#).

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits>::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios<_CharT, _Traits>::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#), and [std::basic_istream<_CharT, _Traits>::unget\(\)](#).

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt (
    const basic_ios<_CharT, _Traits> & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits >::exceptions()`, `std::basic_ios<_CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::eof ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits >::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios< _CharT, _Traits >::exceptions ( ) const [inline], [inherited]
Throwing exceptions on errors.
```

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits >::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::exceptions (
    iostate __except ) [inline], [inherited]
Throwing exceptions on errors.
```

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the

following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
    std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

    std::ifstream f ("/etc/motd");

    std::cerr << "Setting badbit\n";
    f.setstate (std::ios_base::badbit);

    std::cerr << "Setting exception mask\n";
    f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios< _CharT, _Traits >::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::fail ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::operator bool\(\)](#), [std::basic_ios< _CharT, _Traits >::operator!\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), and [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill ( ) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios< _CharT, _Traits >::widen\(\)](#).

Referenced by [std::basic_ios< _CharT, _Traits >::copyfmt\(\)](#), and [std::basic_ios< _CharT, _Traits >::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::fill (
    char_type __ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::__detail::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::flush [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::flush\(\)](#).

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, `std::operator>>()`, and `std::ws()`.

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::good ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References `std::basic_ios<_CharT, _Traits>::rdstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::__detail::operator>>()`.

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios<_CharT, _Traits>::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits>::init (
    basic_streambuf<_CharT, _Traits> * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios<_CharT, _Traits>::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

operator"!()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

References [std::basic_ios< _CharT, _Traits >::fail\(\)](#).

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ostream_type &(*) (__ostream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

operator<<() [3/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    __streambuf_type * __sb ) [inherited]
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

operator<<() [4/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [5/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    const void * __p ) [inline], [inherited]
```

Pointer arithmetic inserters.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    float __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    int __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as std::endl and std::hex use these functions in constructs like "std::cout << std::endl". For more information, see the iomanip header.

operator<<() [10/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_ n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [13/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    short __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

$_ \leftrightarrow$	A variable of builtin integral type.
$_ n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.


References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), and [std::basic_ios< _CharT, _Traits >](#)

operator<<() [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [15/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned short __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

put()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (
    char_type __c ) [inherited]
```

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

*this

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline], [inherited]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__stringbuf_type * std::basic_ostringstream< _CharT, _Traits, _Alloc >::rdbuf ( ) const [inline]
```

Accessing the underlying buffer.

Returns

The current basic_stringbuf buffer.

This hides both signatures of [std::basic_ios::rdbuf\(\)](#).

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_stringbuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_stringbuf< _CharT, _Traits > * __sb ) [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
std::streambuf* p = .....;

foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT, typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seek()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::tellg()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekp() [1/2]

```
template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [2/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (
    pos_type __pos ) [inherited]
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state ) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References `std::basic_ios< _CharT, _Traits >::clear()`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::getline()`, `std::getline()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

str() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__string_type std::basic_ostringstream< _CharT, _Traits, _Alloc >::str ( ) const [inline]
```

Copying out the string buffer.

Returns

`rddbuf() -> str()`

str() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_ostringstream< _CharT, _Traits, _Alloc >::str (
    const __string_type & __s ) [inline]
```

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf() -> str(s)`.

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellp()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp [inherited]
```

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`

tie() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios<_CharT, _Traits>::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream<char, _Traits>::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

write()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (
    const char_type * __s,
    streamsize __n ) [inherited]
```

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.255.6 Member Data Documentation

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put<_CharT, _OutIter>::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`.

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ios<_CharT, _Traits>::bad()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::get()`, `std::basic_ostream<char, _Traits>::operator<<()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator>>()`, `std::basic_istream<char>::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<char>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_istream<_CharT, _Traits>::unget()` and `std::ws()`.

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::hex()`, and `std::oct()`.

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```

Request a seek relative to the beginning of the stream.

Referenced by `std::basic_filebuf<_CharT, _Traits>::seekpos()`.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<char>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::read\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), and [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

fixed

const [fmtflags](#) std::ios_base::fixed [static], [inherited]

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

const [fmtflags](#) std::ios_base::floatfield [static], [inherited]

A mask of scientific|fixed. Useful for the 2-arg form of [setf](#).

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

const [iostate](#) std::ios_base::goodbit [static], [inherited]

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unsetg\(\)](#), and [std::ws\(\)](#).

hex

const [fmtflags](#) std::ios_base::hex [static], [inherited]

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::hex\(\)](#).

in

const [openmode](#) std::ios_base::in [static], [inherited]

Open for input. Default for [ifstream](#) and [fstream](#).

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_filebuf<_CharT, _Traits>::underflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::xsgetn\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by [std::num_put< _CharT, _OutIter >::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::seekp\(\)](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekp\(\)](#), [std::basic_ostream< _CharT, _Traits >::tellp\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

const [fmtflags](#) std::ios_base::skipws [static], [inherited]

Skips leading white space before certain input operations.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

const [openmode](#) std::ios_base::trunc [static], [inherited]

Truncate an existing stream when opening. Default for ofstream.

unitbuf

const [fmtflags](#) std::ios_base::unitbuf [static], [inherited]

Flushes output after each output operation.

Referenced by [std::nunitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

const [fmtflags](#) std::ios_base::uppercase [static], [inherited]

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Referenced by [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

6.256 std::basic_regex< _Ch_type, _Rx_traits > Class Template Reference

```
#include <regex.h>
```

Public Types

- typedef [regex_constants::syntax_option_type](#) **flag_type**
- typedef traits_type::locale_type **locale_type**
- typedef traits_type::string_type **string_type**
- typedef _Rx_traits **traits_type**
- typedef _Ch_type **value_type**

Public Member Functions

- [basic_regex](#) () noexcept
- template<typename _FwdIter >
[basic_regex](#) (_FwdIter __first, _FwdIter __last, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) ([basic_regex](#) &&__rhs) noexcept=default
- [basic_regex](#) (const _Ch_type *__p, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) (const _Ch_type *__p, std::size_t __len, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) (const [basic_regex](#) &__rhs)=default
- template<typename _Ch_traits, typename _Ch_alloc >
[basic_regex](#) (const [std::basic_string](#)< _Ch_type, _Ch_traits, _Ch_alloc > &__s, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) ([initializer_list](#)< _Ch_type > __l, [flag_type](#) __f=ECMAScript)

- `~basic_regex ()`
- `template<typename _InputIterator >
basic_regex & assign (_InputIterator __first, _InputIterator __last, flag_type __flags=ECMAScript)`
- `basic_regex & assign (basic_regex && __rhs) noexcept`
- `basic_regex & assign (const _Ch_type * __p, flag_type __flags=ECMAScript)`
- `basic_regex & assign (const _Ch_type * __p, size_t __len, flag_type __flags=ECMAScript)`
- `basic_regex & assign (const basic_regex & __rhs) noexcept`
- `template<typename _Ch_traits, typename _Alloc >
basic_regex & assign (const basic_string< _Ch_type, _Ch_traits, _Alloc > & __s, flag_type __flags=ECMAScript)`
- `basic_regex & assign (initializer_list< _Ch_type > __l, flag_type __flags=ECMAScript)`
- `flag_type flags () const noexcept`
- `locale_type getloc () const noexcept`
- `locale_type imbue (locale_type __loc)`
- `unsigned int mark_count () const noexcept`
- `basic_regex & operator= (basic_regex &&) = default`
- `basic_regex & operator= (const _Ch_type * __p)`
- `basic_regex & operator= (const basic_regex &) = default`
- `template<typename _Ch_traits, typename _Alloc >
basic_regex & operator= (const basic_string< _Ch_type, _Ch_traits, _Alloc > & __s)`
- `basic_regex & operator= (initializer_list< _Ch_type > __l)`
- `void swap (basic_regex & __rhs) noexcept`

Static Public Attributes

Constants

std [28.8.1](1)

- static constexpr `flag_type icase`
- static constexpr `flag_type nosubs`
- static constexpr `flag_type optimize`
- static constexpr `flag_type collate`
- static constexpr `flag_type ECMAScript`
- static constexpr `flag_type basic`
- static constexpr `flag_type extended`
- static constexpr `flag_type awk`
- static constexpr `flag_type grep`
- static constexpr `flag_type egrep`
- static constexpr `flag_type multiline`

Friends

- `template<typename _Bp, typename _Ap, typename _Cp, typename _Rp >
bool __detail::__regex_algo_impl (_Bp, _Bp, match_results< _Bp, _Ap > &, const basic_regex< _Cp, _Rp > &, regex_constants::match_flag_type, __detail::__RegexExecutorPolicy, bool)`
- `template<typename, typename, typename, bool >
class __detail::__Executor`

Related Symbols

(Note that these are not member symbols.)

- `template<typename _Ch_type, typename _Rx_traits >
void swap (basic_regex< _Ch_type, _Rx_traits > & __lhs, basic_regex< _Ch_type, _Rx_traits > & __rhs) noexcept`

6.256.1 Detailed Description

```
template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>>
class std::basic_regex< _Ch_type, _Rx_traits >
```

Objects of specializations of this class represent regular expressions constructed from sequences of character type `_Ch_type`.

Storage for the regular expression is allocated and deallocated as necessary by the member functions of this class.

6.256.2 Constructor & Destructor Documentation

basic_regex() [1/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex ( ) [inline], [noexcept]
```

Constructs a basic regular expression that does not match any character sequence.

basic_regex() [2/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (
    const _Ch_type * __p,
    flag_type __f = ECMAScript ) [inline], [explicit]
```

Constructs a basic regular expression from the sequence `[__p, __p + char_traits<_Ch_type>::length(__p))` interpreted according to the flags in `__f`.

Parameters

<code>__p</code>	A pointer to the start of a C-style null-terminated string containing a regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<code>regex_error</code>	if <code>__p</code> is not a valid regular expression.
--------------------------	--

basic_regex() [3/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (
    const _Ch_type * __p,
    std::size_t __len,
    flag_type __f = ECMAScript ) [inline]
```

Constructs a basic regular expression from the sequence `[p, p + len)` interpreted according to the flags in `f`.

Parameters

<code>__p</code>	A pointer to the start of a string containing a regular expression.
<code>__len</code>	The length of the string containing the regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<i>regex_error</i>	if <code>__p</code> is not a valid regular expression.
--------------------	--

basic_regex() [4/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (
    const basic_regex< _Ch_type, _Rx_traits > & __rhs ) [default]
```

Copy-constructs a basic regular expression.

Parameters

<code>__rhs</code>	A regex object.
--------------------	-----------------

basic_regex() [5/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (
    basic_regex< _Ch_type, _Rx_traits > && __rhs ) [default], [noexcept]
```

Move-constructs a basic regular expression.

Parameters

<code>__rhs</code>	A regex object.
--------------------	-----------------

basic_regex() [6/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
template<typename _Ch_traits , typename _Ch_alloc >
std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (
    const std::basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s,
    flag_type __f = ECMAScript ) [inline], [explicit]
```

Constructs a basic regular expression from the string `s` interpreted according to the flags in `f`.

Parameters

<code>__s</code>	A string containing a regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<i>regex_error</i>	if <code>__s</code> is not a valid regular expression.
--------------------	--

basic_regex() [7/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
template<typename _FwdIter >
std::basic_regex<_Ch_type, _Rx_traits >::basic_regex (
    _FwdIter __first,
    _FwdIter __last,
    flag_type __f = ECMAScript ) [inline]
```

Constructs a basic regular expression from the range `[first, last)` interpreted according to the flags in `f`.

Parameters

<code>__first</code>	The start of a range containing a valid regular expression.
<code>__last</code>	The end of a range containing a valid regular expression.
<code>__f</code>	The format flags of the regular expression.

Exceptions

<code>regex_error</code>	if <code>[__first, __last)</code> is not a valid regular expression.
--------------------------	--

References `std::basic_regex<_Ch_type, _Rx_traits>::assign()`.

basic_regex() [8/8]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
std::basic_regex<_Ch_type, _Rx_traits >::basic_regex (
    initializer_list<_Ch_type> __l,
    flag_type __f = ECMAScript ) [inline]
```

Constructs a basic regular expression from an initializer list.

Parameters

<code>__l</code>	The initializer list.
<code>__f</code>	The format flags of the regular expression.

Exceptions

<code>regex_error</code>	if <code>__l</code> is not a valid regular expression.
--------------------------	--

~basic_regex()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
```

`std::basic_regex< _Ch_type, _Rx_traits >::~~basic_regex ()` [inline]
 Destroys a basic regular expression.

6.256.3 Member Function Documentation

assign() [1/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
template<typename _InputIterator >
basic_regex & std::basic_regex< _Ch_type, _Rx_traits >::assign (
    _InputIterator __first,
    _InputIterator __last,
    flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object.

Parameters

<code>__first</code>	The start of a range containing a valid regular expression.
<code>__last</code>	The end of a range containing a valid regular expression.
<code>__flags</code>	Syntax option flags.

Exceptions

<code>regex_error</code>	if <code>p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, the object remains unchanged.
--------------------------	--

References `std::basic_regex< _Ch_type, _Rx_traits >::assign()`.

assign() [2/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex< _Ch_type, _Rx_traits >::assign (
    basic_regex< _Ch_type, _Rx_traits > && __rhs ) [inline], [noexcept]
```

Move-assigns one regular expression to another.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

References `std::move()`.

assign() [3/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex< _Ch_type, _Rx_traits >::assign (
    const _Ch_type * __p,
    flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object from a C-style null-terminated string containing a regular expression pattern.

Parameters

<code>__p</code>	A pointer to a C-style null-terminated string containing a regular expression pattern.
------------------	--

Parameters

<code>__flags</code>	Syntax option flags.
----------------------	----------------------

Exceptions

<code>regex_error</code>	if <code>__p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------------	--

assign() [4/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits>::assign (
    const _Ch_type * __p,
    size_t __len,
    flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object from a C-style string containing a regular expression pattern.

Parameters

<code>__p</code>	A pointer to a C-style string containing a regular expression pattern.
<code>__len</code>	The length of the regular expression pattern string.
<code>__flags</code>	Syntax option flags.

Exceptions

<code>regex_error</code>	if <code>p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------------	--

assign() [5/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits>::assign (
    const basic_regex<_Ch_type, _Rx_traits> & __rhs ) [inline], [noexcept]
```

Assigns one regular expression to another.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

Referenced by `std::basic_regex<_Ch_type, _Rx_traits>::basic_regex()`, `std::basic_regex<_Ch_type, _Rx_traits>::assign()`, `std::basic_regex<_Ch_type, _Rx_traits>::operator=()`, `std::basic_regex<_Ch_type, _Rx_traits>::operator=()`, and `std::basic_regex<_Ch_type, _Rx_traits>::operator=()`.

assign() [6/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
template<typename _Ch_traits , typename _Alloc >
basic_regex & std::basic_regex<_Ch_type, _Rx_traits>::assign (
```

```
const basic_string< _Ch_type, _Ch_traits, _Alloc > & __s,
flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object from a string containing a regular expression pattern.

Parameters

<code>__s</code>	A string containing a regular expression pattern.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>__s</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------	--

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

assign() [7/7]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex< _Ch_type, _Rx_traits >::assign (
    initializer_list< _Ch_type > __l,
    flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object.

Parameters

<code>__l</code>	An initializer list representing a regular expression.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>__l</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, the object remains unchanged.
--------------------	--

flags()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
flag_type std::basic_regex< _Ch_type, _Rx_traits >::flags ( ) const [inline], [noexcept]
```

Gets the flags used to construct the regular expression or in the last call to `assign()`.

getloc()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
locale_type std::basic_regex< _Ch_type, _Rx_traits >::getloc ( ) const [inline], [noexcept]
```

Gets the locale currently imbued in the regular expression object.

imbue()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
locale_type std::basic_regex< _Ch_type, _Rx_traits >::imbue (
```

```
locale_type __loc ) [inline]
```

Imbues the regular expression object with the given locale.

Parameters

<code>__loc</code>	A locale.
--------------------	-----------

References [std::swap\(\)](#).

mark_count()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
unsigned int std::basic_regex<_Ch_type, _Rx_traits >::mark_count ( ) const [inline], [noexcept]
```

Gets the number of marked subexpressions within the regular expression.

operator=() [1/5]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits >::operator= (
    basic_regex<_Ch_type, _Rx_traits > && ) [default]
```

Move-assigns one regular expression to another.

operator=() [2/5]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits >::operator= (
    const _Ch_type * __p ) [inline]
```

Replaces a regular expression with a new one constructed from a C-style null-terminated string.

Parameters

<code>__p</code>	A pointer to the start of a null-terminated C-style string containing a regular expression.
------------------	---

References [std::basic_regex<_Ch_type, _Rx_traits >::assign\(\)](#).

operator=() [3/5]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits >::operator= (
    const basic_regex<_Ch_type, _Rx_traits > & ) [default]
```

Assigns one regular expression to another.

operator=() [4/5]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
template<typename _Ch_traits , typename _Alloc >
basic_regex & std::basic_regex<_Ch_type, _Rx_traits >::operator= (
    const basic_string<_Ch_type, _Ch_traits, _Alloc > & __s ) [inline]
```

Replaces a regular expression with a new one constructed from a string.

Parameters

<code>__s</code>	A pointer to a string containing a regular expression.
------------------	--

References [std::basic_regex<_Ch_type, _Rx_traits >::assign\(\)](#).

operator=() [5/5]

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
basic_regex & std::basic_regex<_Ch_type, _Rx_traits >::operator= (
    initializer_list<_Ch_type > __l ) [inline]
```

Replaces a regular expression with a new one constructed from an initializer list.

Parameters

↵	The initializer list.
↵	
↵	
↵	
↵	

Exceptions

<i>regex_error</i>	if <code>__l</code> is not a valid regular expression.
--------------------	--

References [std::basic_regex<_Ch_type, _Rx_traits >::assign\(\)](#).

swap()

```
template<typename _Ch_type , typename _Rx_traits = regex_traits<_Ch_type>>
void std::basic_regex<_Ch_type, _Rx_traits >::swap (
    basic_regex<_Ch_type, _Rx_traits > & __rhs ) [inline], [noexcept]
```

Swaps the contents of two regular expression objects.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

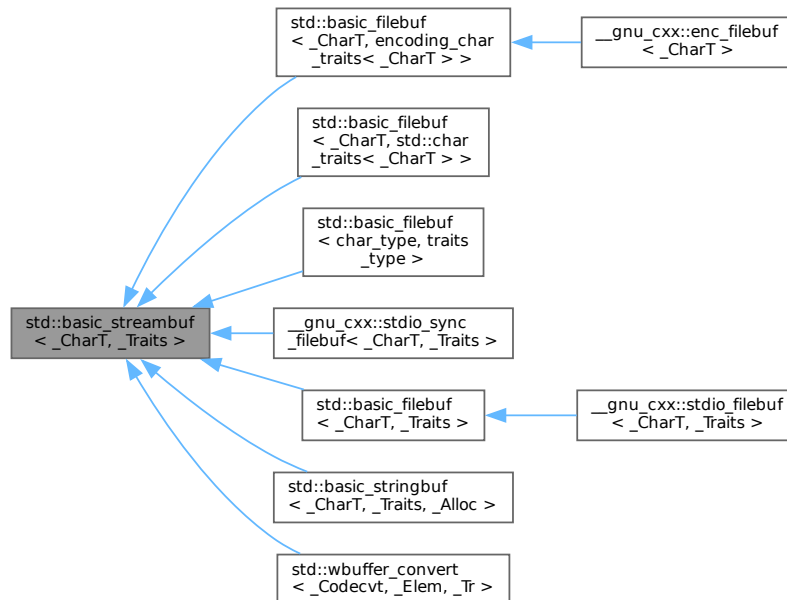
References [std::swap\(\)](#).

The documentation for this class was generated from the following file:

- [regex.h](#)

6.257 std::basic_streambuf< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::basic_streambuf< _CharT, _Traits >:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Traits` `traits_type`
- typedef `traits_type::int_type` `int_type`
- typedef `traits_type::pos_type` `pos_type`
- typedef `traits_type::off_type` `off_type`
- typedef `basic_streambuf< char_type, traits_type >` `__streambuf_type`

Public Member Functions

- virtual `~basic_streambuf()`
- `locale getloc()` const
- `streamsize in_avail()`
- `locale pubimbue(const locale &__loc)`
- `int_type sbumpc()`
- `int_type sgetc()`
- `streamsize sgetn(char_type * __s, streamsize __n)`
- `int_type snextc()`
- `int_type sputbackc(char_type __c)`
- `int_type sputc(char_type __c)`

- [streamsize sputn](#) (const [char_type](#) *__s, [streamsize](#) __n)
- [int_type sungetc](#) ()
- [basic_streambuf](#) * [pubsetbuf](#) ([char_type](#) *__s, [streamsize](#) __n)
- [pos_type](#) [pubseekoff](#) ([off_type](#) __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- [pos_type](#) [pubseekpos](#) ([pos_type](#) __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- [int](#) [pubsync](#) ()

Protected Member Functions

- [basic_streambuf](#) ()
- **[basic_streambuf](#)** (const [basic_streambuf](#) &)
- void **[__safe_gbump](#)** ([streamsize](#) __n)
- void **[__safe_pbump](#)** ([streamsize](#) __n)
- void [gbump](#) (int __n)
- virtual void [imbue](#) (const [locale](#) &__loc)
- [basic_streambuf](#) & **[operator=](#)** (const [basic_streambuf](#) &)
- virtual [int_type](#) [overflow](#) ([int_type](#) __c=[traits_type::eof\(\)](#))
- virtual [int_type](#) [pbackfail](#) ([int_type](#) __c=[traits_type::eof\(\)](#))
- void [pbump](#) (int __n)
- virtual [pos_type](#) [seekoff](#) ([off_type](#), [ios_base::seekdir](#), [ios_base::openmode](#)=[ios_base::in](#)|[ios_base::out](#))
- virtual [pos_type](#) [seekpos](#) ([pos_type](#), [ios_base::openmode](#)=[ios_base::in](#)|[ios_base::out](#))
- virtual [basic_streambuf](#)< [char_type](#), [_Traits](#) > * [setbuf](#) ([char_type](#) *, [streamsize](#))
- void [setg](#) ([char_type](#) *__gbeg, [char_type](#) *__gnext, [char_type](#) *__gend)
- void [setp](#) ([char_type](#) *__pbeg, [char_type](#) *__pend)
- virtual [streamsize](#) [showmanyc](#) ()
- void **[swap](#)** ([basic_streambuf](#) &__sb)
- virtual [int](#) [sync](#) ()
- virtual [int_type](#) [uflow](#) ()
- virtual [int_type](#) [underflow](#) ()
- virtual [streamsize](#) [xsgetn](#) ([char_type](#) *__s, [streamsize](#) __n)
- virtual [streamsize](#) [xspurn](#) (const [char_type](#) *__s, [streamsize](#) __n)
- [char_type](#) * [eback](#) () const
- [char_type](#) * [gptr](#) () const
- [char_type](#) * [egptr](#) () const
- [char_type](#) * [pbase](#) () const
- [char_type](#) * [pptr](#) () const
- [char_type](#) * [epptr](#) () const

Protected Attributes

- [locale](#) [_M_buf_locale](#)
- [char_type](#) * [_M_in_beg](#)
- [char_type](#) * [_M_in_cur](#)
- [char_type](#) * [_M_in_end](#)
- [char_type](#) * [_M_out_beg](#)
- [char_type](#) * [_M_out_cur](#)
- [char_type](#) * [_M_out_end](#)

Friends

- template<bool _IsMove, typename _CharT2 >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, _CharT2 * >::__type __copy_move_a2 (istreambuf_iterator< _CharT2 >, istreambuf_iterator< _CharT2 >, _CharT2 *)
- streamsize __copy_streambufs_eof (basic_streambuf *, basic_streambuf *, bool &)
- void __istream_extract (istream &, char *, streamsize)
- template<typename _CharT2, typename _Distance >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, void >::__type advance (istreambuf_iterator< _CharT2 > &, _Distance)
- class basic_ios< char_type, traits_type >
- class basic_istream< char_type, traits_type >
- class basic_ostream< char_type, traits_type >
- template<typename _CharT2 >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, istreambuf_iterator< _CharT2 > >::__type find (istreambuf_iterator< _CharT2 >, istreambuf_iterator< _CharT2 >, const _CharT2 &)
- template<typename _CharT2, typename _Traits2, typename _Alloc >
basic_istream< _CharT2, _Traits2 > & getline (basic_istream< _CharT2, _Traits2 > &, basic_string< _CharT2, _Traits2, _Alloc > &, _CharT2)
- class istreambuf_iterator< char_type, traits_type >
- template<typename _CharT2, typename _Traits2, typename _Alloc >
basic_istream< _CharT2, _Traits2 > & operator>> (basic_istream< _CharT2, _Traits2 > &, basic_string< _CharT2, _Traits2, _Alloc > &)
- class ostreambuf_iterator< char_type, traits_type >

6.257.1 Detailed Description

template<typename _CharT, typename _Traits>
class std::basic_streambuf< _CharT, _Traits >

The actual work of input and output (interface).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This is a base class. Derived stream buffers each control a pair of character sequences: one for input, and one for output.

Section [27.5.1] of the standard describes the requirements and behavior of stream buffer classes. That section (three paragraphs) is reproduced here, for simplicity and accuracy.

1. Stream buffers can impose various constraints on the sequences they control. Some constraints are:

- The controlled input sequence can be not readable.
- The controlled output sequence can be not writable.
- The controlled sequences can be associated with the contents of other representations for character sequences, such as external files.
- The controlled sequences can support operations *directly* to or from associated sequences.
- The controlled sequences can impose limitations on how the program can read characters from a sequence, write characters to a sequence, put characters back into an input sequence, or alter the stream position.

2. Each sequence is characterized by three pointers which, if non-null, all point into the same `charT` array object. The array object represents, at any moment, a (sub)sequence of characters from the sequence. Operations

performed on a sequence alter the values stored in these pointers, perform reads and writes directly to or from associated sequences, and alter *the stream position* and conversion state as needed to maintain this subsequence relationship. The three pointers are:

- the *beginning pointer*, or lowest element address in the array (called *xbeg* here);
- the *next pointer*, or next element address that is a current candidate for reading or writing (called *xnext* here);
- the *end pointer*, or first element address beyond the end of the array (called *xend* here).

3. The following semantic constraints shall always apply for any set of three pointers for a sequence, using the pointer names given immediately above:

- If *xnext* is not a null pointer, then *xbeg* and *xend* shall also be non-null pointers into the same `charT` array, as described above; otherwise, *xbeg* and *xend* shall also be null.
- If *xnext* is not a null pointer and *xnext* < *xend* for an output sequence, then a *write position* is available. In this case, **xnext* shall be assignable as the next element to write (to put, or to store a character value, into the sequence).
- If *xnext* is not a null pointer and *xbeg* < *xnext* for an input sequence, then a *putback position* is available. In this case, *xnext*[-1] shall have a defined value and is the next (preceding) element to store a character that is put back into the input sequence.
- If *xnext* is not a null pointer and *xnext* < *xend* for an input sequence, then a *read position* is available. In this case, **xnext* shall have a defined value and is the next element to read (to get, or to obtain a character value, from the sequence).

6.257.2 Member Typedef Documentation

`__streambuf_type`

```
template<typename _CharT , typename _Traits >
typedef basic_streambuf<char_type, traits_type> std::basic_streambuf< _CharT, _Traits >::__↵
streambuf_type
```

This is a non-standard type.

`char_type`

```
template<typename _CharT , typename _Traits >
typedef _CharT std::basic_streambuf< _CharT, _Traits >::char_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

`int_type`

```
template<typename _CharT , typename _Traits >
typedef traits_type::int_type std::basic_streambuf< _CharT, _Traits >::int_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

`off_type`

```
template<typename _CharT , typename _Traits >
typedef traits_type::off_type std::basic_streambuf< _CharT, _Traits >::off_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

pos_type

```
template<typename _CharT , typename _Traits >
typedef traits_type::pos_type std::basic_streambuf< _CharT, _Traits >::pos_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

traits_type

```
template<typename _CharT , typename _Traits >
typedef _Traits std::basic_streambuf< _CharT, _Traits >::traits_type
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

6.257.3 Constructor & Destructor Documentation**~basic_streambuf()**

```
template<typename _CharT , typename _Traits >
virtual std::basic_streambuf< _CharT, _Traits >::~~basic_streambuf ( ) [inline], [virtual]
```

Destructor deallocates no buffer space.

basic_streambuf()

```
template<typename _CharT , typename _Traits >
std::basic_streambuf< _CharT, _Traits >::basic_streambuf ( ) [inline], [protected]
```

Base constructor.

Only called from derived constructors, and sets up all the buffer data to zero, including the pointers described in the basic_streambuf class description. Note that, as a result,

- the class starts with no read nor write positions available,
- this is not an error

6.257.4 Member Function Documentation**eback()**

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::egptr ( ) const [inline], [protected]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence

- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow\(\)](#).

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline]
```

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow\(\)](#).

imbue()

```
template<typename _CharT , typename _Traits >
virtual void std::basic_streambuf< _CharT, _Traits >::imbue (
    const locale & __loc ) [inline], [protected], [virtual]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

in_avail()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::in_avail ( ) [inline]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

overflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::overflow (
    int_type __c = traits_type::eof() ) [inline], [protected], [virtual]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns eof().

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, `__gnu_cxx::stdio_sync_filebuf<std::basic_stringbuf<_CharT, _Traits, _Alloc>`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>`.

pbackfail()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf<_CharT, _Traits>::pbackfail (
    int_type __c = traits_type::eof() ) [inline], [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

eof() on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns eof().

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, `__gnu_cxx::stdio_sync_filebuf<std::basic_stringbuf<_CharT, _Traits, _Alloc>`.

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf<_CharT, _Traits>::pbase ( ) const [inline], [protected]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf<_CharT, _Traits>::pbump (
    int __n ) [inline], [protected]
```

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- epptr() returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline]
```

Calls virtual `sync` function.

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline]
```

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Referenced by `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< char >::ignore()`, `std::istreambuf_iterator< _CharT, _Traits >::operator++()`, and `std::basic_istream< char >::seekg()`.

seekoff()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits>](#).

seekpos()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits>](#).

setbuf()

```
template<typename _CharT , typename _Traits >
virtual basic_streambuf< char_type, _Traits > * std::basic_streambuf< _CharT, _Traits >::setbuf
(
    char_type * ,
    streamsize ) [inline], [protected], [virtual]
```

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits>](#).

setg()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend ) [inline], [protected]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

setp()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setp (
    char_type * __pbeg,
    char_type * __pend ) [inline], [protected]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == ep_ptr()`

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::istreambuf_iterator<_CharT, _Traits>::operator++\(\)](#), and [std::basic_istream<char>::seekg\(\)](#).

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
    char_type * __s,
    streamsize __n ) [inline]
```

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsggetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
template<typename _CharT, typename _Traits>
virtual streamsize std::basic_streambuf<_CharT, _Traits>::showmanyc ( ) [inline], [protected],
[virtual]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, _Traits>`.

snextc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::snextc ( ) [inline]
```

Getting the next character.

Returns

The next character, or eof.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::seekg()`, and `std::basic_istream<char>::unget()`.

sputbackc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
    char_type __c ) [inline]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sputc (
    char_type __c ) [inline]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::sputn (
    const char_type * __s,
    streamsize __n ) [inline]
```

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xputn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

sungetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sungetc ( ) [inline]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

sync()

```
template<typename _CharT , typename _Traits >
virtual int std::basic_streambuf< _CharT, _Traits >::sync ( ) [inline], [protected], [virtual]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented in `std::wbuffer_convert<_Codecvt, _Elem, _Tr>`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

uflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

underflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::underflow ( ) [inline], [protected],
[virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns eof().

Reimplemented in [std::wbuffer_convert<_Codecvt, _Elem, _Tr>](#), [__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#).

xsgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::xsgetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

References [std::min\(\)](#).

xsputn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::xsputn (
    const char_type * __s,
    streamsize __n ) [protected], [virtual]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

References `std::min()`.

6.257.5 Member Data Documentation

`_M_buf_locale`

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale [protected]
```

Current locale setting.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

`_M_in_beg`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg [protected]
```

Start of get area.

`_M_in_cur`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur [protected]
```

Current read area.

`_M_in_end`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end [protected]
```

End of get area.

`_M_out_beg`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg [protected]
```

Start of put area.

`_M_out_cur`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur [protected]
```

Current put area.

`_M_out_end`

```
template<typename _CharT , typename _Traits >
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end [protected]
```

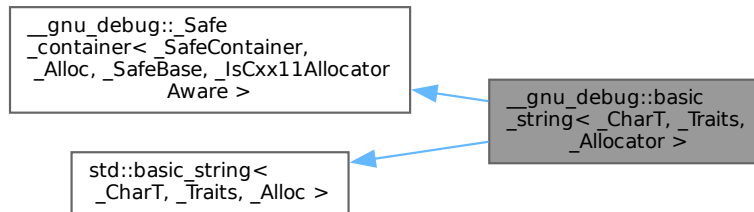
End of put area.

The documentation for this class was generated from the following files:

- `iosfwd`
- `streambuf`
- `streambuf.tcc`

6.258 `__gnu_debug::basic_string<_CharT, _Traits, _Allocator>` Class Template Reference

Inheritance diagram for `__gnu_debug::basic_string<_CharT, _Traits, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator<typename _Base::const_iterator, basic_string>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator<const_iterator>` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::_Safe_iterator<typename _Base::iterator, basic_string>` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator<iterator>` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Traits` **traits_type**
- typedef `_Traits::char_type` **value_type**

Public Member Functions

- **basic_string** (`_Base` && `__base`) noexcept
- template<typename `_InputIterator`>
 basic_string (`_InputIterator` `__begin`, `_InputIterator` `__end`, const `_Allocator` & `__a`=`_Allocator`())
- **basic_string** (`basic_string` &&)=default
- **basic_string** (`basic_string` && `__s`, const `_Allocator` & `__a`) noexcept(`std::is_nothrow_constructible<_Base, _Base, const _Allocator &>::value`)
- **basic_string** (const `_Allocator` & `__a`) noexcept
- **basic_string** (const `_Base` & `__base`)
- **basic_string** (const `_CharT` * `__s`, const `_Allocator` & `__a`=`_Allocator`())
- **basic_string** (const `_CharT` * `__s`, `size_type` `__n`, const `_Allocator` & `__a`=`_Allocator`())
- **basic_string** (const `basic_string` &)=default
- **basic_string** (const `basic_string` & `__s`, const `_Allocator` & `__a`)
- **basic_string** (const `basic_string` & `__str`, `size_type` `__pos`, `size_type` `__n`=`_Base::npos`, const `_Allocator` & `__a`=`_Allocator`())
- **basic_string** (`size_type` `__n`, `_CharT` `__c`, const `_Allocator` & `__a`=`_Allocator`())
- **basic_string** (`std::initializer_list<_CharT>` `__l`, const `_Allocator` & `__a`=`_Allocator`())

- `const _Base & _M_base ()` `const noexcept`
- `_Base & _M_base ()` `noexcept`
- `template<typename _InputIterator >`
`basic_string<_CharT, _Traits, _Alloc> & _M_replace_dispatch (iterator __i1, iterator __i2, _InputIterator __k1,`
`_InputIterator __k2, __false_type)`
- `template<typename _InIterator >`
`_CharT * _S_construct (_InIterator __beg, _InIterator __end, const _Alloc & __a, forward_iterator_tag)`
- `template<typename _InputIterator >`
`basic_string & append (_InputIterator __first, _InputIterator __last)`
- `basic_string & append (const _CharT * __s)`
- `basic_string & append (const _CharT * __s, size_type __n)`
- `template<typename _Tp >`
`_If_sv<_Tp, basic_string & > append (const _Tp & __svt)`
- `template<typename _Tp >`
`_If_sv<_Tp, basic_string & > append (const _Tp & __svt, size_type __pos, size_type __n=npos)`
- `basic_string & append (const basic_string & __str)`
- `basic_string & append (const basic_string & __str)`
- `basic_string & append (const basic_string & __str, size_type __pos, size_type __n)`
- `basic_string & append (const basic_string & __str, size_type __pos, size_type __n=npos)`
- `basic_string & append (initializer_list<_CharT> __l)`
- `basic_string & append (size_type __n, _CharT __c)`
- `template<typename _InputIterator >`
`basic_string & assign (_InputIterator __first, _InputIterator __last)`
- `basic_string & assign (basic_string && __str) noexcept(allocator_traits<_Alloc>::is_always_equal::value)`
- `basic_string & assign (basic_string && __x) noexcept(noexcept(std::declval<_Base & >().assign(std::move(__x)))`
`__x)))`
- `basic_string & assign (const _CharT * __s)`
- `basic_string & assign (const _CharT * __s, size_type __n)`
- `template<typename _Tp >`
`_If_sv<_Tp, basic_string & > assign (const _Tp & __svt)`
- `template<typename _Tp >`
`_If_sv<_Tp, basic_string & > assign (const _Tp & __svt, size_type __pos, size_type __n=npos)`
- `basic_string & assign (const basic_string & __str)`
- `basic_string & assign (const basic_string & __str, size_type __pos, size_type __n)`
- `basic_string & assign (const basic_string & __str, size_type __pos, size_type __n=npos)`
- `basic_string & assign (const basic_string & __x)`
- `basic_string & assign (size_type __n, _CharT __c)`
- `basic_string & assign (std::initializer_list<_CharT> __l)`
- `reference at (size_type __n)`
- `const_reference at (size_type __n) const`
- `reference back ()`
- `const_reference back () const noexcept`
- `iterator begin ()`
- `const_iterator begin () const noexcept`
- `const _CharT * c_str () const noexcept`
- `size_type capacity () const noexcept`
- `const_iterator cbegin () const noexcept`
- `const_iterator cend () const noexcept`
- `void clear ()`
- `int compare (const _CharT * __s) const`
- `template<typename _Tp >`
`_If_sv<_Tp, int > compare (const _Tp & __svt) const noexcept(is_same<_Tp, __sv_type>::value)`

- int **compare** (const **basic_string** &__str) const
- int **compare** (const **basic_string** &__str) const
- template<typename _Tp >
_If_sv< _Tp, int > **compare** (size_type __pos, size_type __n, const _Tp &__svt) const noexcept(is_same< _Tp, __sv_type >::value)
- int **compare** (size_type __pos, size_type __n, const **basic_string** &__str) const
- int **compare** (size_type __pos1, size_type __n1, const _CharT * __s) const
- int **compare** (size_type __pos1, size_type __n1, const _CharT * __s, size_type __n2) const
- template<typename _Tp >
_If_sv< _Tp, int > **compare** (size_type __pos1, size_type __n1, const _Tp &__svt, size_type __pos2, size_type __n2=**npos**) const noexcept(is_same< _Tp, __sv_type >::value)
- int **compare** (size_type __pos1, size_type __n1, const **basic_string** &__str) const
- int **compare** (size_type __pos1, size_type __n1, const **basic_string** &__str, size_type __pos2, size_type __n2) const
- int **compare** (size_type __pos1, size_type __n1, const **basic_string** &__str, size_type __pos2, size_type __n2=**npos**) const
- size_type **copy** (_CharT * __s, size_type __n, size_type __pos=0) const
- **const_reverse_iterator** **crbegin** () const noexcept
- **const_reverse_iterator** **crend** () const noexcept
- const _CharT * **data** () const noexcept
- _CharT * **data** () noexcept
- bool **empty** () const noexcept
- **iterator** **end** ()
- **const_iterator** **end** () const noexcept
- bool **ends_with** (_CharT __x) const noexcept
- bool **ends_with** (basic_string_view< _CharT, _Traits > __x) const noexcept
- bool **ends_with** (const _CharT * __x) const noexcept
- **iterator** **erase** (__const_iterator __first, __const_iterator __last)
- **iterator** **erase** (__const_iterator __position)
- **iterator** **erase** (iterator __first, iterator __last)
- **iterator** **erase** (iterator __position)
- **basic_string** & **erase** (size_type __pos=0, size_type __n=**Base::npos**)
- size_type **find** (_CharT __c, size_type __pos=0) const noexcept
- size_type **find** (const _CharT * __s, size_type __pos, size_type __n) const
- size_type **find** (const _CharT * __s, size_type __pos=0) const
- template<typename _Tp >
_If_sv< _Tp, size_type > **find** (const _Tp &__svt, size_type __pos=0) const noexcept(is_same< _Tp, __sv_type >::value)
- size_type **find** (const **basic_string** &__str, size_type __pos=0) const noexcept
- size_type **find** (const **basic_string** &__str, size_type __pos=0) const noexcept
- size_type **find_first_not_of** (_CharT __c, size_type __pos=0) const noexcept
- size_type **find_first_not_of** (const _CharT * __s, size_type __pos, size_type __n) const
- size_type **find_first_not_of** (const _CharT * __s, size_type __pos=0) const
- template<typename _Tp >
_If_sv< _Tp, size_type > **find_first_not_of** (const _Tp &__svt, size_type __pos=0) const noexcept(is_same< _Tp, __sv_type >::value)
- size_type **find_first_not_of** (const **basic_string** &__str, size_type __pos=0) const noexcept
- size_type **find_first_not_of** (const **basic_string** &__str, size_type __pos=0) const noexcept
- size_type **find_first_of** (_CharT __c, size_type __pos=0) const noexcept
- size_type **find_first_of** (const _CharT * __s, size_type __pos, size_type __n) const
- size_type **find_first_of** (const _CharT * __s, size_type __pos=0) const

- `template<typename _Tp>`
`_If_sv<_Tp, size_type> find_first_of` (const `_Tp` &__svt, size_type __pos=0) const noexcept(is_same<_Tp, __sv_type>::value)
- size_type `find_first_of` (const `basic_string` &__str, size_type __pos=0) const noexcept
- size_type `find_first_of` (const `basic_string` &__str, size_type __pos=0) const noexcept
- size_type `find_last_not_of` (_CharT __c, size_type __pos= `Base::npos`) const noexcept
- size_type `find_last_not_of` (const `_CharT` *__s, size_type __pos, size_type __n) const
- size_type `find_last_not_of` (const `_CharT` *__s, size_type __pos= `Base::npos`) const
- `template<typename _Tp>`
`_If_sv<_Tp, size_type> find_last_not_of` (const `_Tp` &__svt, size_type __pos= `npos`) const noexcept(is_same<_Tp, __sv_type>::value)
- size_type `find_last_not_of` (const `basic_string` &__str, size_type __pos= `Base::npos`) const noexcept
- size_type `find_last_not_of` (const `basic_string` &__str, size_type __pos= `npos`) const noexcept
- size_type `find_last_of` (_CharT __c, size_type __pos= `Base::npos`) const noexcept
- size_type `find_last_of` (const `_CharT` *__s, size_type __pos, size_type __n) const
- size_type `find_last_of` (const `_CharT` *__s, size_type __pos= `Base::npos`) const
- `template<typename _Tp>`
`_If_sv<_Tp, size_type> find_last_of` (const `_Tp` &__svt, size_type __pos= `npos`) const noexcept(is_same<_Tp, __sv_type>::value)
- size_type `find_last_of` (const `basic_string` &__str, size_type __pos= `Base::npos`) const noexcept
- size_type `find_last_of` (const `basic_string` &__str, size_type __pos= `npos`) const noexcept
- reference `front` ()
- const_reference `front` () const noexcept
- allocator_type `get_allocator` () const noexcept
- iterator `insert` (__const_iterator __p, _CharT __c)
- `template<typename _InputIterator>`
`iterator insert` (__const_iterator __p, _InputIterator __first, _InputIterator __last)
- `iterator insert` (const_iterator __p, size_type __n, _CharT __c)
- `iterator insert` (const_iterator __p, std::initializer_list<_CharT> __l)
- `iterator insert` (iterator __p, _CharT __c)
- `template<class _InputIterator>`
`void insert` (iterator __p, _InputIterator __beg, _InputIterator __end)
- `void insert` (iterator __p, initializer_list<_CharT> __l)
- `void insert` (iterator __p, size_type __n, _CharT __c)
- `basic_string` & `insert` (size_type __pos, const `_CharT` *__s)
- `basic_string` & `insert` (size_type __pos, const `_CharT` *__s, size_type __n)
- `template<typename _Tp>`
`_If_sv<_Tp, basic_string &> insert` (size_type __pos, const `_Tp` &__svt)
- `basic_string` & `insert` (size_type __pos, size_type __n, _CharT __c)
- `template<typename _Tp>`
`_If_sv<_Tp, basic_string &> insert` (size_type __pos1, const `_Tp` &__svt, size_type __pos2, size_type __n= `npos`)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str, size_type __pos2, size_type __n)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str, size_type __pos2, size_type __n= `npos`)
- size_type `length` () const noexcept
- size_type `max_size` () const noexcept
- operator `__sv_type` () const noexcept
- `basic_string` & `operator+=` (_CharT __c)
- `basic_string` & `operator+=` (const `_CharT` *__s)

- `template<typename _Tp >`
`_If_sv< _Tp, basic_string & > operator+= (const _Tp &__svt)`
- `basic_string & operator+= (const basic_string &__str)`
- `basic_string & operator+= (const basic_string &__str)`
- `basic_string & operator+= (std::initializer_list< _CharT > __l)`
- `basic_string & operator= (_CharT __c)`
- `basic_string & operator= (basic_string &&)=default`
- `basic_string & operator= (const _CharT *__s)`
- `basic_string & operator= (const basic_string &)=default`
- `basic_string & operator= (std::initializer_list< _CharT > __l)`
- `reference operator[] (size_type __pos)`
- `const_reference operator[] (size_type __pos) const noexcept`
- `void pop_back ()`
- `void push_back (_CharT __c)`
- `reverse_iterator rbegin ()`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend ()`
- `const_reverse_iterator rend () const noexcept`
- `template<typename _InputIterator >`
`basic_string & replace (__const_iterator __i1, __const_iterator __i2, _InputIterator __j1, _InputIterator __j2)`
- `basic_string & replace (__const_iterator __i1, __const_iterator __i2, const _CharT *__s)`
- `basic_string & replace (__const_iterator __i1, __const_iterator __i2, const _CharT *__s, size_type __n)`
- `basic_string & replace (__const_iterator __i1, __const_iterator __i2, const basic_string &__str)`
- `basic_string & replace (__const_iterator __i1, __const_iterator __i2, size_type __n, _CharT __c)`
- `basic_string & replace (__const_iterator __i1, __const_iterator __i2, std::initializer_list< _CharT > __l)`
- `template<typename _Tp >`
`_If_sv< _Tp, basic_string & > replace (const_iterator __i1, const_iterator __i2, const _Tp &__svt)`
- `basic_string & replace (iterator __i1, iterator __i2, _CharT *__k1, _CharT *__k2)`
- `template<class _InputIterator >`
`basic_string & replace (iterator __i1, iterator __i2, _InputIterator __k1, _InputIterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__k1, const _CharT *__k2)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__s)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__s, size_type __n)`
- `basic_string & replace (iterator __i1, iterator __i2, const basic_string &__str)`
- `basic_string & replace (iterator __i1, iterator __i2, const_iterator __k1, const_iterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, initializer_list< _CharT > __l)`
- `basic_string & replace (iterator __i1, iterator __i2, iterator __k1, iterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, size_type __n, _CharT __c)`
- `template<typename _Tp >`
`_If_sv< _Tp, basic_string & > replace (size_type __pos, size_type __n, const _Tp &__svt)`
- `basic_string & replace (size_type __pos, size_type __n, const basic_string &__str)`
- `basic_string & replace (size_type __pos, size_type __n1, const _CharT *__s)`
- `basic_string & replace (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2)`
- `basic_string & replace (size_type __pos, size_type __n1, size_type __n2, _CharT __c)`
- `template<typename _Tp >`
`_If_sv< _Tp, basic_string & > replace (size_type __pos1, size_type __n1, const _Tp &__svt, size_type __pos2, size_type __n2=npos)`
- `basic_string & replace (size_type __pos1, size_type __n1, const basic_string &__str)`
- `basic_string & replace (size_type __pos1, size_type __n1, const basic_string &__str, size_type __pos2, size_type __n2)`

- `basic_string` & `replace` (size_type __pos1, size_type __n1, const `basic_string` & __str, size_type __pos2, size_type __n2=`npos`)
- void `reserve` ()
- void `reserve` (size_type __res_arg)
- void `resize` (size_type __n)
- void `resize` (size_type __n, _CharT __c)
- size_type `rfind` (_CharT __c, size_type __pos=`Base::npos`) const noexcept
- size_type `rfind` (const _CharT * __s, size_type __pos, size_type __n) const
- size_type `rfind` (const _CharT * __s, size_type __pos=`Base::npos`) const
- template<typename _Tp >
_If_sv< _Tp, size_type > `rfind` (const _Tp & __svt, size_type __pos=`npos`) const noexcept(is_same< _Tp, __sv_type >::value)
- size_type `rfind` (const `basic_string` & __str, size_type __pos=`Base::npos`) const noexcept
- size_type `rfind` (const `basic_string` & __str, size_type __pos=`npos`) const noexcept
- void `shrink_to_fit` () noexcept
- size_type `size` () const noexcept
- bool `starts_with` (_CharT __x) const noexcept
- bool `starts_with` (basic_string_view< _CharT, _Traits > __x) const noexcept
- bool `starts_with` (const _CharT * __x) const noexcept
- `basic_string` `substr` (size_type __pos=0, size_type __n=`Base::npos`) const
- void `swap` (`basic_string` & __s) noexcept(/*conditional */)
- void `swap` (`basic_string` & __x) noexcept(/*conditional */)

Static Public Attributes

- static const size_type `npos`

Protected Member Functions

- void `_M_swap` (_Safe_container & __x) noexcept

Friends

- template<typename _ItT, typename _SeqT, typename _CatT >
class ::`__gnu_debug::Safe_iterator`

6.258.1 Detailed Description

template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>

class `__gnu_debug::basic_string<_CharT, _Traits, _Allocator>`

Class std::basic_string with safety/checking/debug instrumentation.

6.258.2 Member Function Documentation

append() [1/5]

```
template<typename _CharT, typename _Traits, typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string<_CharT, _Traits, _Alloc >::append (
    const _Tp & __svt ) [inline], [inherited]
```

Append a string_view.

Parameters

<code>__svt</code>	The object convertible to <code>string_view</code> to be appended.
--------------------	--

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#).

append() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::append (
    const _Tp & __svt,
    size_type __pos,
    size_type __n = npos ) [inline], [inherited]
```

Append a range of characters from a `string_view`.

Parameters

<code>__svt</code>	The object convertible to <code>string_view</code> to be appended from.
<code>__pos</code>	The position in the <code>string_view</code> to append from.
<code>__n</code>	The number of characters to append from the <code>string_view</code> .

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#).

append() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic\_string< _CharT, _Traits, _Alloc > & std::basic\_string< _CharT, _Traits, _Alloc >::append (
    const basic\_string< _CharT, _Traits, _Alloc > & __str ) [inherited]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#), and [std::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::collate<_CharT>::do_transform\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator+=\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator+=\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator+=\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator+=\(\)](#), and [std::operator>>\(\)](#).

append() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::append (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos,
    size_type __n = npos ) [inherited]
```

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of <code>str</code> to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

References `std::size()`.

append() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::append (
    initializer_list< _CharT > __l ) [inline], [inherited]
```

Append an `initializer_list` of characters.

Parameters

<code>↵</code>	The <code>initializer_list</code> of characters to append.
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code>/</code>	

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`.

assign() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    basic_string< _CharT, _Traits, _Alloc > && __str ) [inline], [noexcept], [inherited]
```


Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

References [std::basic_string<_CharT, _Traits, _Alloc>::swap\(\)](#).

assign() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::assign (
    const _Tp & __svt ) [inline], [inherited]
```

Set value from a string_view.

Parameters

<code>__svt</code>	The source object convertible to string_view.
--------------------	---

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#).

assign() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::assign (
    const _Tp & __svt,
    size_type __pos,
    size_type __n = npos ) [inline], [inherited]
```

Set value from a range of characters in a string_view.

Parameters

<code>__svt</code>	The source object convertible to string_view.
<code>__pos</code>	The position in the string_view to assign from.
<code>__n</code>	The number of characters to assign.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#).

assign() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inherited]
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

References [std::basic_string< _CharT, _Traits, _Alloc >::get_allocator\(\)](#).

Referenced by [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::operator=\(\)](#), and [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#).

assign() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos,
    size_type __n = npos ) [inline], [inherited]
```

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.

References [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#).

at() [1/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
reference std::basic_string< _CharT, _Traits, _Alloc >::at (
    size_type __n ) [inline]
```

Provides access to the data contained in the string.

Parameters

\leftrightarrow <code>_n</code>	The index of the character to access.
--------------------------------------	---------------------------------------

Returns

Read/write reference to the character.

Exceptions

<code>std::out_of_range</code>	If <i>n</i> is an invalid index.
--------------------------------	----------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails. Success results in unsharing the string.

at() [2/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
const_reference std::basic_string< _CharT, _Traits, _Alloc >::at (
    size_type __n ) const [inline]
```

Provides access to the data contained in the string.

Parameters

\leftrightarrow <code>_n</code>	The index of the character to access.
--------------------------------------	---------------------------------------

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If <i>n</i> is an invalid index.
--------------------------------	----------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.

back() [1/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
reference std::basic_string< _CharT, _Traits, _Alloc >::back ( ) [inline]
```

Returns a read/write reference to the data at the last element of the string.

back() [2/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
const_reference std::basic_string< _CharT, _Traits, _Alloc >::back ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the last element of the string.

capacity()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
size_type std::basic\_string<_CharT, _Traits, _Alloc >::capacity ( ) const [inline], [noexcept]
```

Returns the total number of characters that the string can hold before needing to allocate more memory.

compare() [1/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, int > std::basic\_string<_CharT, _Traits, _Alloc >::compare (
    const _Tp & __svt ) const [inline], [noexcept], [inherited]
```

Compare to a `string_view`.

Parameters

<code>__svt</code>	An object convertible to <code>string_view</code> to compare against.
--------------------	---

Returns

Integer < 0 , 0 , or > 0 .

References [std::min\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

compare() [2/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic\_string<_CharT, _Traits, _Alloc >::compare (
    const basic\_string<_CharT, _Traits, _Alloc > & __str ) const [inline], [inherited]
```

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0 , 0 , or > 0 .

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::min\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

compare() [3/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, int > std::basic\_string<_CharT, _Traits, _Alloc >::compare (
    size_type __pos,
```

```

        size_type __n,
        const _Tp & __svt ) const [inline], [noexcept], [inherited]

```

Compare to a `string_view`.

Parameters

<code>__pos</code>	A position in the string to start comparing from.
<code>__n</code>	The number of characters to compare.
<code>__svt</code>	An object convertible to <code>string_view</code> to compare against.

Returns

Integer < 0 , 0 , or > 0 .

`compare()` [4/6]

```

template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos,
    size_type __n,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) const [inherited]

```

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(),str.data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::min()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

`compare()` [5/6]

```

template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, int > std::basic_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos1,
    size_type __n1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n2 = npos ) const [inline], [noexcept], [inherited]

```

Compare to a `string_view`.

Parameters

<code>__pos1</code>	A position in the string to start comparing from.
<code>__n1</code>	The number of characters to compare.
<code>__svt</code>	An object convertible to <code>string_view</code> to compare against.
<code>__pos2</code>	A position in the <code>string_view</code> to start comparing from.
<code>__n2</code>	The number of characters to compare.

Returns

Integer < 0, 0, or > 0.

compare() [6/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string<_CharT, _Traits, _Alloc>::compare (
    size_type __pos1,
    size_type __n1,
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos2,
    size_type __n2 = npos ) const [inherited]
```

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of str.
<code>__n2</code>	Number of characters in substring of str.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(), str.substr(pos2, n2).data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::min()`.

data()

```
template<typename _CharT , typename _Traits , typename _Alloc >
_CharT * std::basic_string<_CharT, _Traits, _Alloc>::data ( ) [inline], [noexcept], [inherited]
```

Return non-const pointer to contents.

This is a pointer to the character sequence held by the string. Modifying the characters in the sequence is allowed.

empty()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
```



```
std::allocator<_CharT>>
bool std::basic_string< _CharT, _Traits, _Alloc >::empty ( ) const [inline], [noexcept]
Returns true if the string is empty. Equivalent to *this == "".
```

erase() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::iterator std::basic_string< _CharT, _Traits, _Alloc >↵
::erase (
    iterator __first,
    iterator __last ) [inherited]
```

Remove a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to remove.
<code>__last</code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range [first,last) from this string. The value of the string doesn't change if an error is thrown.

erase() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
iterator std::basic_string< _CharT, _Traits, _Alloc >::erase (
    iterator __position ) [inline], [inherited]
```

Remove one character.

Parameters

<code>__position</code>	Iterator referencing the character to remove.
-------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

find() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic_string< _CharT, _Traits, _Alloc >::find (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#).

find() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::find (
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

find_first_not_of() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv<_Tp, size_type> std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a character not in a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#).

find_first_not_of() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of (
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#).

find_first_of() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic_string< _CharT, _Traits, _Alloc >::find_first_of (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a character of a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#).

find_first_of() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_of (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = 0 ) const [inline], [noexcept], [inherited]
```

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, and `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`.

find_last_not_of() [1/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, size_type> std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a character not in a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

References `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`.

find_last_not_of() [2/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
size_type std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of (
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, and `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`.

find_last_of() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic\_string< _CharT, _Traits, _Alloc >::find_last_of (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a character of string.

Parameters

<code>__svt</code>	An object convertible to <code>string_view</code> containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

References [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#).

find_last_of() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string< _CharT, _Traits, _Alloc >::find_last_of (
    const basic\_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

Referenced by [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#).

front() [1/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
reference std::basic\_string< _CharT, _Traits, _Alloc >::front ( ) [inline]
```

Returns a read/write reference to the data at the first element of the string.

front() [2/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

const_reference `std::basic_string<_CharT, _Traits, _Alloc>::front () const` [inline], [noexcept]
Returns a read-only (constant) reference to the data at the first element of the string.

`get_allocator()`

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
allocator_type std::basic_string<_CharT, _Traits, _Alloc>::get_allocator ( ) const [inline],
[noexcept]
```

Return copy of allocator used to construct this string.

`insert()` [1/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
iterator std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    _CharT __c ) [inline], [inherited]
```

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

`insert()` [2/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<class _InputIterator>
void std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    _InputIterator __beg,
    _InputIterator __end ) [inline], [inherited]
```

Insert a range of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts characters in range `[__beg, __end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

insert() [3/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
void std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    initializer_list<_CharT> __l ) [inline], [inherited]
```

Insert an initializer_list of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__l</code>	The initializer_list of characters to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [4/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
void std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    size_type __n,
    _CharT __c ) [inline], [inherited]
```

Insert multiple characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [5/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, basic_string &> std::basic_string<_CharT, _Traits, _Alloc>::insert (
    size_type __pos,
    const _Tp & __svt ) [inline], [inherited]
```

Insert a string_view.

Parameters

<code>__pos</code>	Position in string to insert at.
<code>__svt</code>	The object convertible to string_view to insert.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [6/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, basic_string &> std::basic_string<_CharT, _Traits, _Alloc>::insert (
    size_type __pos1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n = npos ) [inline], [inherited]
```

Insert a string_view.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__svt</code>	The object convertible to string_view to insert from.
<code>__pos2</code>	Position in string_view to insert from.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

insert() [7/8]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::insert (
```



```
size_type __pos1,
const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline], [inherited]
```

Insert value of a string.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `std::basic_string< _CharT, _Traits, _Alloc >::insert()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

`insert()` [8/8]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos1,
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos2,
    size_type __n = npos ) [inline], [inherited]
```

Insert a substring.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>pos1 > size()</code> or <code>__pos2 > str.size()</code> .

Starting at `pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

length()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

```
size_type std::basic\_string<\_CharT, \_Traits, \_Alloc>::length ( ) const [inline], [noexcept]
```

Returns the number of characters in the string, not including any null-termination.

max_size()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

```
size_type std::basic\_string<\_CharT, \_Traits, \_Alloc>::max\_size ( ) const [inline], [noexcept]
```

Returns the size() of the largest possible string.

operator __sv_type()

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
std::basic\_string<\_CharT, \_Traits, \_Alloc>::operator \_\_sv\_type ( ) const [inline], [noexcept],
[inherited]
```

Convert to a string_view.

Returns

A string_view.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

operator+=() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
template<typename _Tp >
```

```
_If_sv< _Tp, basic\_string & > std::basic\_string<\_CharT, \_Traits, \_Alloc>::operator+= (
    const _Tp & __svt ) [inline], [inherited]
```

Append a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to be appended.
--------------------	---

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#).

operator+=() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
basic\_string & std::basic\_string<\_CharT, \_Traits, \_Alloc>::operator+= (
    const basic\_string<\_CharT, \_Traits, \_Alloc> & __str ) [inline], [inherited]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#).

replace() [1/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::replace (
    const_iterator __i1,
    const_iterator __i2,
    const _Tp & __svt ) [inline], [inherited]
```

Replace range of characters with `string_view`.

Parameters

<code>__i1</code>	An iterator referencing the start position to replace at.
<code>__i2</code>	An iterator referencing the end position for the replace.
<code>__svt</code>	The object convertible to <code>string_view</code> to insert from.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::begin\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [2/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<class _InputIterator >
basic\_string & std::basic\_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    _InputIterator __k1,
    _InputIterator __k2 ) [inline], [inherited]
```

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, characters in the range `[__k1,__k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [3/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    const _CharT * __s ) [inline], [inherited]
```

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [4/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    const _CharT * __s,
    size_type __n ) [inline], [inherited]
```

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from s to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the first `__n` characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [5/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    iterator __i1,
    iterator __i2,
    const basic_string<_CharT, _Traits, _Alloc> & __str ) [inline], [inherited]
```

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the value of `__str` is inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

replace() [6/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    iterator __i1,
    iterator __i2,
    initializer_list<_CharT> __l ) [inline], [inherited]
```

Replace range of characters with `initializer_list`.

Parameters

<code>↵ __i1</code>	Iterator referencing start of range to replace.
-------------------------	---

Parameters

<code>__i2</code>	Iterator referencing end of range to replace.
<code>__l</code>	The initializer_list of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, characters in the range `[__k1, __k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [7/11]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    iterator __i1,
    iterator __i2,
    size_type __n,
    _CharT __c) [inline], [inherited]
```

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

[std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#),
[std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#),
[std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#),
[std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#),
[std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [10/11]

```

template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, basic_string &> std::basic_string<_CharT, _Traits, _Alloc>::replace (
    size_type __pos1,
    size_type __n1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n2 = npos ) [inline], [inherited]
  
```

Replace range of characters with `string_view`.

Parameters

<code>__pos1</code>	The position to replace at.
<code>__n1</code>	The number of characters to replace.
<code>__svt</code>	The object convertible to <code>string_view</code> to insert from.
<code>__pos2</code>	The position in the <code>string_view</code> to insert from.
<code>__n2</code>	The number of characters to insert.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [11/11]

```

template<typename _CharT, typename _Traits, typename _Alloc>
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    size_type __pos1,
    size_type __n1,
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos2,
    size_type __n2 = npos ) [inline], [inherited]
  
```

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of <code>str</code> to use.
<code>__n2</code>	Number of characters from <code>str</code> to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos1, __pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

reserve() [1/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

```
void std::basic_string< _CharT, _Traits, _Alloc >::reserve
```

Equivalent to `shrink_to_fit()`.

reserve() [2/2]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

```
void std::basic_string< _CharT, _Traits, _Alloc >::reserve (
    size_type __res_arg )
```

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<code>std::length_error</code>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

rfind() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
template<typename _Tp >
```

```
_If_sv< _Tp, size_type > std::basic_string< _CharT, _Traits, _Alloc >::rfind (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a string_view.

Parameters

<code>__svt</code>	The object convertible to <code>string_view</code> to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#).

rfind() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::rfind (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = npos ) const [inline], [noexcept], [inherited]
```

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#).

size()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
```

```
size_type std::basic_string< _CharT, _Traits, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the number of characters in the string, not including any null-termination.

swap()

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::swap (
    basic_string< _CharT, _Traits, _Alloc > & __s ) [noexcept], [inherited]
```

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::assign()`, and `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`.

6.258.3 Member Data Documentation

npos

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator =
std::allocator<_CharT>>
const basic_string<_CharT, _Traits, _Alloc>::size_type std::basic_string<_CharT, _Traits, _↵
Alloc>::npos [static]
```

Value returned by various member functions when they fail.

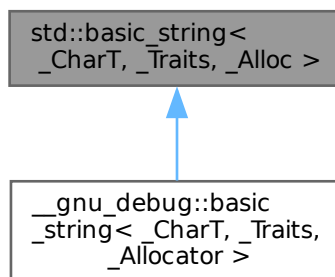
The documentation for this class was generated from the following file:

- [debug/string](#)

6.259 std::basic_string<_CharT, _Traits, _Alloc> Class Template Reference

```
#include <string>
```

Inheritance diagram for `std::basic_string<_CharT, _Traits, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `__gnu_cxx::__normal_iterator< const_pointer, basic_string >` **const_iterator**
- typedef `_CharT_alloc_traits::const_pointer` **const_pointer**
- typedef `const value_type &` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_CharT_alloc_traits::difference_type` **difference_type**
- typedef `__gnu_cxx::__normal_iterator< pointer, basic_string >` **iterator**
- typedef `_CharT_alloc_traits::pointer` **pointer**
- typedef `value_type &` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_CharT_alloc_traits::size_type` **size_type**
- typedef `_Traits` **traits_type**
- typedef `_Traits::char_type` **value_type**

Public Member Functions

- [basic_string](#) () noexcept
- [template<class _InputIterator >](#)
[basic_string](#) (_InputIterator __beg, _InputIterator __end, const _Alloc &__a=_Alloc())
- [basic_string](#) ([basic_string](#) &&__str) noexcept
- [basic_string](#) ([basic_string](#) &&__str, const _Alloc &__a)
- [basic_string](#) (const _Alloc &__a)
- [template<typename = _RequireAllocator<_Alloc>>](#)
[basic_string](#) (const _CharT *__s, const _Alloc &__a=_Alloc())
- [basic_string](#) (const _CharT *__s, size_type __n, const _Alloc &__a=_Alloc())
- [template<typename _Tp, typename = _If_sv<_Tp, void>>](#)
[basic_string](#) (const _Tp &__t, const _Alloc &__a=_Alloc())
- [template<typename _Tp, typename = enable_if_t<is_convertible_v<const _Tp&, __sv_type>>>](#)
[basic_string](#) (const _Tp &__t, size_type __pos, size_type __n, const _Alloc &__a=_Alloc())
- [basic_string](#) (const [basic_string](#) &__str)
- [basic_string](#) (const [basic_string](#) &__str, const _Alloc &__a)
- [basic_string](#) (const [basic_string](#) &__str, size_type __pos, const _Alloc &__a=_Alloc())
- [basic_string](#) (const [basic_string](#) &__str, size_type __pos, size_type __n)
- [basic_string](#) (const [basic_string](#) &__str, size_type __pos, size_type __n, const _Alloc &__a)
- [basic_string](#) (initializer_list<_CharT> __l, const _Alloc &__a=_Alloc())
- [basic_string](#) (size_type __n, _CharT __c, const _Alloc &__a=_Alloc())
- [~basic_string](#) () noexcept
- [template<typename _InputIterator >](#)
[basic_string](#)<_CharT, _Traits, _Alloc> & [_M_replace_dispatch](#) (iterator __i1, iterator __i2, _InputIterator __k1, _InputIterator __k2, __false_type)
- [template<typename _InIterator >](#)
_CharT * [_S_construct](#) (_InIterator __beg, _InIterator __end, const _Alloc &__a, [forward_iterator_tag](#))
- [template<class _InputIterator >](#)
[basic_string](#) & [append](#) (_InputIterator __first, _InputIterator __last)
- [basic_string](#) & [append](#) (const _CharT *__s)
- [basic_string](#) & [append](#) (const _CharT *__s, size_type __n)
- [template<typename _Tp >](#)
_If_sv<_Tp, [basic_string](#) &> [append](#) (const _Tp &__svt)
- [template<typename _Tp >](#)
_If_sv<_Tp, [basic_string](#) &> [append](#) (const _Tp &__svt, size_type __pos, size_type __n=[npos](#))
- [basic_string](#) & [append](#) (const [basic_string](#) &__str)
- [basic_string](#) & [append](#) (const [basic_string](#) &__str, size_type __pos, size_type __n=[npos](#))
- [basic_string](#) & [append](#) (initializer_list<_CharT> __l)
- [basic_string](#) & [append](#) (size_type __n, _CharT __c)
- [template<class _InputIterator >](#)
[basic_string](#) & [assign](#) (_InputIterator __first, _InputIterator __last)
- [basic_string](#) & [assign](#) ([basic_string](#) &&__str) noexcept([allocator_traits](#)<_Alloc>::is_always_equal::value)
- [basic_string](#) & [assign](#) (const _CharT *__s)
- [basic_string](#) & [assign](#) (const _CharT *__s, size_type __n)
- [template<typename _Tp >](#)
_If_sv<_Tp, [basic_string](#) &> [assign](#) (const _Tp &__svt)
- [template<typename _Tp >](#)
_If_sv<_Tp, [basic_string](#) &> [assign](#) (const _Tp &__svt, size_type __pos, size_type __n=[npos](#))
- [basic_string](#) & [assign](#) (const [basic_string](#) &__str)
- [basic_string](#) & [assign](#) (const [basic_string](#) &__str, size_type __pos, size_type __n=[npos](#))
- [basic_string](#) & [assign](#) (initializer_list<_CharT> __l)

- `basic_string` & `assign` (size_type __n, _CharT __c)
- reference `at` (size_type __n)
- const_reference `at` (size_type __n) const
- reference `back` ()
- const_reference `back` () const noexcept
- iterator `begin` ()
- const_iterator `begin` () const noexcept
- const _CharT * `c_str` () const noexcept
- size_type `capacity` () const noexcept
- const_iterator `cbegin` () const noexcept
- const_iterator `cend` () const noexcept
- void `clear` () noexcept
- int `compare` (const _CharT *__s) const noexcept
- template<typename _Tp >
_If_sv< _Tp, int > `compare` (const _Tp &__svt) const noexcept(`is_same`< _Tp, __sv_type >::value)
- int `compare` (const `basic_string` &__str) const
- template<typename _Tp >
_If_sv< _Tp, int > `compare` (size_type __pos, size_type __n, const _Tp &__svt) const noexcept(`is_same`< _Tp, __sv_type >::value)
- int `compare` (size_type __pos, size_type __n, const `basic_string` &__str) const
- int `compare` (size_type __pos, size_type __n1, const _CharT *__s) const
- int `compare` (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2) const
- template<typename _Tp >
_If_sv< _Tp, int > `compare` (size_type __pos1, size_type __n1, const _Tp &__svt, size_type __pos2, size_type __n2=`npos`) const noexcept(`is_same`< _Tp, __sv_type >::value)
- int `compare` (size_type __pos1, size_type __n1, const `basic_string` &__str, size_type __pos2, size_type __n2=`npos`) const
- size_type `copy` (_CharT *__s, size_type __n, size_type __pos=0) const
- const_reverse_iterator `crbegin` () const noexcept
- const_reverse_iterator `crend` () const noexcept
- const _CharT * `data` () const noexcept
- _CharT * `data` () noexcept
- bool `empty` () const noexcept
- iterator `end` ()
- const_iterator `end` () const noexcept
- bool `ends_with` (_CharT __x) const noexcept
- bool `ends_with` (`basic_string_view`< _CharT, _Traits > __x) const noexcept
- bool `ends_with` (const _CharT *__x) const noexcept
- iterator `erase` (iterator __first, iterator __last)
- iterator `erase` (iterator __position)
- `basic_string` & `erase` (size_type __pos=0, size_type __n=`npos`)
- size_type `find` (_CharT __c, size_type __pos=0) const noexcept
- size_type `find` (const _CharT *__s, size_type __pos, size_type __n) const noexcept
- size_type `find` (const _CharT *__s, size_type __pos=0) const noexcept
- template<typename _Tp >
_If_sv< _Tp, size_type > `find` (const _Tp &__svt, size_type __pos=0) const noexcept(`is_same`< _Tp, __sv_type >::value)
- size_type `find` (const `basic_string` &__str, size_type __pos=0) const noexcept
- size_type `find_first_not_of` (_CharT __c, size_type __pos=0) const noexcept
- size_type `find_first_not_of` (const _CharT *__s, size_type __pos, size_type __n) const noexcept
- size_type `find_first_not_of` (const _CharT *__s, size_type __pos=0) const noexcept

- template<typename _Tp >
_If_sv< _Tp, size_type > [find_first_not_of](#) (const _Tp &__svt, size_type __pos=0) const noexcept([is_same](#)< _Tp, __sv_type >::value)
- size_type [find_first_not_of](#) (const [basic_string](#) &__str, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (const _CharT * __s, size_type __pos, size_type __n) const noexcept
- size_type [find_first_of](#) (const _CharT * __s, size_type __pos=0) const noexcept
- template<typename _Tp >
_If_sv< _Tp, size_type > [find_first_of](#) (const _Tp &__svt, size_type __pos=0) const noexcept([is_same](#)< _Tp, __sv_type >::value)
- size_type [find_first_of](#) (const [basic_string](#) &__str, size_type __pos=0) const noexcept
- size_type [find_last_not_of](#) (_CharT __c, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_not_of](#) (const _CharT * __s, size_type __pos, size_type __n) const noexcept
- size_type [find_last_not_of](#) (const _CharT * __s, size_type __pos=[npos](#)) const noexcept
- template<typename _Tp >
_If_sv< _Tp, size_type > [find_last_not_of](#) (const _Tp &__svt, size_type __pos=[npos](#)) const noexcept([is_same](#)< _Tp, __sv_type >::value)
- size_type [find_last_not_of](#) (const [basic_string](#) &__str, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_of](#) (_CharT __c, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_of](#) (const _CharT * __s, size_type __pos, size_type __n) const noexcept
- size_type [find_last_of](#) (const _CharT * __s, size_type __pos=[npos](#)) const noexcept
- template<typename _Tp >
_If_sv< _Tp, size_type > [find_last_of](#) (const _Tp &__svt, size_type __pos=[npos](#)) const noexcept([is_same](#)< _Tp, __sv_type >::value)
- size_type [find_last_of](#) (const [basic_string](#) &__str, size_type __pos=[npos](#)) const noexcept
- reference [front](#) ()
- const_reference [front](#) () const noexcept
- allocator_type [get_allocator](#) () const noexcept
- iterator [insert](#) (iterator __p, _CharT __c)
- template<class _InputIterator >
void [insert](#) (iterator __p, _InputIterator __beg, _InputIterator __end)
- void [insert](#) (iterator __p, [initializer_list](#)< _CharT > __l)
- void [insert](#) (iterator __p, size_type __n, _CharT __c)
- [basic_string](#) & [insert](#) (size_type __pos, const _CharT * __s)
- [basic_string](#) & [insert](#) (size_type __pos, const _CharT * __s, size_type __n)
- template<typename _Tp >
_If_sv< _Tp, [basic_string](#) & > [insert](#) (size_type __pos, const _Tp & __svt)
- [basic_string](#) & [insert](#) (size_type __pos, size_type __n, _CharT __c)
- template<typename _Tp >
_If_sv< _Tp, [basic_string](#) & > [insert](#) (size_type __pos1, const _Tp & __svt, size_type __pos2, size_type __n=[npos](#))
- [basic_string](#) & [insert](#) (size_type __pos1, const [basic_string](#) & __str)
- [basic_string](#) & [insert](#) (size_type __pos1, const [basic_string](#) & __str, size_type __pos2, size_type __n=[npos](#))
- size_type [length](#) () const noexcept
- size_type [max_size](#) () const noexcept
- operator __sv_type () const noexcept
- [basic_string](#) & operator+= (_CharT __c)
- [basic_string](#) & operator+= (const _CharT * __s)
- template<typename _Tp >
_If_sv< _Tp, [basic_string](#) & > operator+= (const _Tp & __svt)
- [basic_string](#) & operator+= (const [basic_string](#) & __str)

- `basic_string & operator+=(initializer_list<_CharT> __l)`
- `basic_string & operator=(_CharT __c)`
- `basic_string & operator=(basic_string &&__str) noexcept(/*conditional */)`
- `basic_string & operator=(const _CharT *__s)`
- `template<typename _Tp>
_If_sv<_Tp, basic_string &> operator=(const _Tp &__svt)`
- `basic_string & operator=(const basic_string &__str)`
- `basic_string & operator=(initializer_list<_CharT> __l)`
- `reference operator[] (size_type __pos)`
- `const_reference operator[] (size_type __pos) const noexcept`
- `void pop_back ()`
- `void push_back (_CharT __c)`
- `reverse_iterator rbegin ()`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend ()`
- `const_reverse_iterator rend () const noexcept`
- `template<typename _Tp>
_If_sv<_Tp, basic_string &> replace (const_iterator __i1, const_iterator __i2, const _Tp &__svt)`
- `basic_string & replace (iterator __i1, iterator __i2, _CharT *__k1, _CharT *__k2)`
- `template<class _InputIterator>
basic_string & replace (iterator __i1, iterator __i2, _InputIterator __k1, _InputIterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__k1, const _CharT *__k2)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__s)`
- `basic_string & replace (iterator __i1, iterator __i2, const _CharT *__s, size_type __n)`
- `basic_string & replace (iterator __i1, iterator __i2, const basic_string &__str)`
- `basic_string & replace (iterator __i1, iterator __i2, const_iterator __k1, const_iterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, initializer_list<_CharT> __l)`
- `basic_string & replace (iterator __i1, iterator __i2, iterator __k1, iterator __k2)`
- `basic_string & replace (iterator __i1, iterator __i2, size_type __n, _CharT __c)`
- `template<typename _Tp>
_If_sv<_Tp, basic_string &> replace (size_type __pos, size_type __n, const _Tp &__svt)`
- `basic_string & replace (size_type __pos, size_type __n, const basic_string &__str)`
- `basic_string & replace (size_type __pos, size_type __n1, const _CharT *__s)`
- `basic_string & replace (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2)`
- `basic_string & replace (size_type __pos, size_type __n1, size_type __n2, _CharT __c)`
- `template<typename _Tp>
_If_sv<_Tp, basic_string &> replace (size_type __pos1, size_type __n1, const _Tp &__svt, size_type __pos2, size_type __n2=npos)`
- `basic_string & replace (size_type __pos1, size_type __n1, const basic_string &__str, size_type __pos2, size_type __n2=npos)`
- `void reserve ()`
- `void reserve (size_type __res_arg)`
- `void resize (size_type __n)`
- `void resize (size_type __n, _CharT __c)`
- `size_type rfind (_CharT __c, size_type __pos=npos) const noexcept`
- `size_type rfind (const _CharT *__s, size_type __pos, size_type __n) const noexcept`
- `size_type rfind (const _CharT *__s, size_type __pos=npos) const noexcept`
- `template<typename _Tp>
_If_sv<_Tp, size_type> rfind (const _Tp &__svt, size_type __pos=npos) const noexcept(is_same<_Tp, __sv_type>::value)`
- `size_type rfind (const basic_string &__str, size_type __pos=npos) const noexcept`

- void `shrink_to_fit` () noexcept
- size_type `size` () const noexcept
- bool `starts_with` (_CharT __x) const noexcept
- bool `starts_with` (basic_string_view< _CharT, _Traits > __x) const noexcept
- bool `starts_with` (const _CharT *__x) const noexcept
- basic_string substr (size_type __pos=0, size_type __n=npos) const
- void `swap` (basic_string &__s) noexcept(*/*conditional*/*)

Static Public Attributes

- static const size_type `npos`

Protected Types

- typedef iterator `__const_iterator`

6.259.1 Detailed Description

template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_string< _CharT, _Traits, _Alloc >

Managing sequences of characters and character-like objects.

Template Parameters

<code>_CharT</code>	Type of character
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#). Of the [optional sequence requirements](#), only `push_back`, `at`, and array access are supported.

Todo Needs documentation! See <http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation+style.html>

Documentation? What's that? Nathan Myers ncm@cantrip.org.

A string looks like this:

```

[basic_string<char_type>]      [_Rep]
                               _M_length
_M_dataplus                   _M_capacity
_M_p ----->                 _M_refcount
                               unnamed array of char_type
```

Where the `_M_p` points to the first character in the string, and you cast it to a pointer-to-`_Rep` and subtract 1 to get a pointer to the header.

This approach has the enormous advantage that a string object requires only one allocation. All the ugliness is confined within a single pair of inline functions, which each compile to a single *add* instruction: `_Rep::_M_data()`, and `string↔::_M_rep()`; and the allocation function which gets a block of raw bytes and with room enough and constructs a `_Rep` object at the front.

The reason you want `_M_data` pointing to the character array and not the `_Rep` is so that the debugger can see the string contents. (Probably we should add a non-inline member to get the `_Rep` for the debugger to use, so users can check the actual string length.)

Note that the `_Rep` object is a POD so that you can have a static *empty string* `_Rep` object already *constructed* before static constructors have run. The reference-count encoding is chosen so that a 0 indicates one reference, so you never try to destroy the empty-string `_Rep` object.

All but the last paragraph is considered pretty conventional for a Copy-On-Write C++ string implementation.

6.259.2 Constructor & Destructor Documentation

basic_string() [1/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string ( ) [inline], [noexcept]
```

Default constructor creates an empty string.

Referenced by [std::basic_string< _CharT, _Traits, _Alloc >::substr\(\)](#).

basic_string() [2/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const _Alloc & __a ) [inline], [explicit]
```

Construct an empty string using allocator *a*.

basic_string() [3/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Construct string with copy of value of *str*.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

basic_string() [4/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos,
    const _Alloc & __a = _Alloc() )
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__a</code>	Allocator to use.

basic_string() [5/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos,
    size_type __n )
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy.

basic_string() [6/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos,
    size_type __n,
    const _Alloc & __a )
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use.

basic_string() [7/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const _CharT * __s,
    size_type __n,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string initialized by a character array.

Parameters

<code>__s</code>	Source character array.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use (default is default allocator).



NB: `__s` must have at least `__n` characters, `'\0'` has no special meaning.

basic_string() [8/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename = _RequireAllocator<_Alloc>>
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const _CharT * __s,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a C string.

Parameters




 <code>_s</code>	Source C string.
 <code>_a</code>	Allocator to use (default is default allocator).

basic_string() [9/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    size_type __n,
    _CharT __c,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as multiple characters.

Parameters

 <code>_n</code>	Number of characters.
 <code>_c</code>	Character to use.
 <code>_a</code>	Allocator to use (default is default allocator).

basic_string() [10/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    basic_string< _CharT, _Traits, _Alloc > && __str ) [inline], [noexcept]
```

Move construct string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------



The newly-created string contains the exact contents of `__str`. `__str` is a valid, but unspecified string.

basic_string() [11/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    initializer_list< _CharT > __l,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string from an initializer list.

Parameters

 <code>_l</code>	std::initializer_list of characters.
 <code>_a</code>	Allocator to use (default is default allocator).

basic_string() [12/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<class _InputIterator >
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    _InputIterator __beg,
    _InputIterator __end,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a range.

Parameters

<code>__beg</code>	Start of range.
<code>__end</code>	End of range.
<code>__a</code>	Allocator to use (default is default allocator).

basic_string() [13/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp , typename = enable_if_t<is_convertible_v<const _Tp&, __sv_type>>>
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const _Tp & __t,
    size_type __pos,
    size_type __n,
    const _Alloc & __a = _Alloc() ) [inline]
```

Construct string from a substring of a string_view.

Parameters

<code>__t</code>	Source object convertible to string view.
<code>__pos</code>	The index of the first character to copy from <code>__t</code> .
<code>__n</code>	The number of characters to copy from <code>__t</code> .
<code>__a</code>	Allocator to use.

basic_string() [14/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp , typename = _If_sv<_Tp, void>>
std::basic_string< _CharT, _Traits, _Alloc >::basic_string (
    const _Tp & __t,
    const _Alloc & __a = _Alloc() ) [inline], [explicit]
```

Construct string from a string_view.

Parameters

<code>↔ __t</code>	Source object convertible to string view.
<code>↔ __a</code>	Allocator to use (default is default allocator).

~basic_string()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string< _CharT, _Traits, _Alloc >::~~basic_string ( ) [inline], [noexcept]
```

Destroy the string instance.

References [std::basic_string<_CharT, _Traits, _Alloc >::get_allocator\(\)](#).

6.259.3 Member Function Documentation**append() [1/9]**

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<class _InputIterator >
basic_string & std::basic_string<_CharT, _Traits, _Alloc >::append (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Append a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Appends characters in the range [`__first`,`__last`) to this string.

References [std::basic_string<_CharT, _Traits, _Alloc >::replace\(\)](#).

append() [2/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc >::append (
    const _CharT * __s ) [inline]
```

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc >::append\(\)](#).

append() [3/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc > & std::basic_string<_CharT, _Traits, _Alloc >::append (
    const _CharT * __s,
    size_type __n )
```

Append a C substring.

Parameters

<code>_↔ _s</code>	The C string to append.
<code>_↔ _n</code>	The number of characters to append.

Returns

Reference to this string.

References [std::size\(\)](#).

append() [4/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::append (
    const _Tp & __svt ) [inline]
```

Append a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to be appended.
--------------------	---

Returns

Reference to this string.

References [std::basic_string< _CharT, _Traits, _Alloc >::append\(\)](#).

append() [5/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::append (
    const _Tp & __svt,
    size_type __pos,
    size_type __n = npos ) [inline]
```

Append a range of characters from a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to be appended from.
<code>__pos</code>	The position in the string_view to append from.
<code>__n</code>	The number of characters to append from the string_view.

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`.

append() [6/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc> & std::basic_string<_CharT, _Traits, _Alloc>::append (
    const basic_string<_CharT, _Traits, _Alloc> & __str )
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::size()`, and `std::size()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::collate<_CharT>::do_transform()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator+=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator+=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator+=()`, and `std::operator>>()`.

append() [7/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc> & std::basic_string<_CharT, _Traits, _Alloc>::append (
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos,
    size_type __n = npos )
```

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of str to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Sets value of string to characters in the range `[__first,__last)`.

References `std::basic_string<_CharT, _Traits, _Alloc>::replace()`.

assign() [2/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    basic_string< _CharT, _Traits, _Alloc > && __str ) [inline], [noexcept]
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

References `std::basic_string<_CharT, _Traits, _Alloc>::swap()`.

assign() [3/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    const _CharT * __s ) [inline]
```

Set value to contents of a C string.

Parameters

<code>__s</code>	The C string to use.
------------------	----------------------

Returns

Reference to this string.

This function sets the value of this string to the value of `__s`. The data is copied, so there is no dependence on `__s` once the function returns.

References `std::basic_string<_CharT, _Traits, _Alloc>::assign()`.

assign() [4/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::assign (
```

```
const _CharT * __s,
size_type __n )
```

Set value to a C substring.

Parameters

<code>__s</code>	The C string to use.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

This function sets the value of this string to the first `__n` characters of `__s`. If `__n` is larger than the number of available characters in `__s`, the remainder of `__s` is used.

References [std::size\(\)](#).

assign() [5/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string< _CharT, _Traits, _Alloc >::assign (
    const _Tp & __svt ) [inline]
```

Set value from a string_view.

Parameters

<code>__svt</code>	The source object convertible to string_view.
--------------------	---

Returns

Reference to this string.

References [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#).

assign() [6/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string< _CharT, _Traits, _Alloc >::assign (
    const _Tp & __svt,
    size_type __pos,
    size_type __n = npos ) [inline]
```

Set value from a range of characters in a string_view.

Parameters

<code>__svt</code>	The source object convertible to string_view.
<code>__pos</code>	The position in the string_view to assign from.
<code>__n</code>	The number of characters to assign.

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::assign()`.

assign() [7/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc> & std::basic_string<_CharT, _Traits, _Alloc>::assign (
    const basic_string<_CharT, _Traits, _Alloc> & __str )
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::get_allocator()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::assign()`, `std::basic_string<_CharT, _Traits, _Alloc>::assign()`, `std::basic_string<_CharT, _Traits, _Alloc>::assign()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator=()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`.

assign() [8/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::assign (
    const basic_string<_CharT, _Traits, _Alloc> & __str,
    size_type __pos,
    size_type __n = npos ) [inline]
```

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.






References [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#).

assign() [9/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    initializer_list< _CharT > __l ) [inline]
```

Set value to an initializer_list of characters.

Parameters

	The initializer_list of characters to assign.
	
	
	
	

Returns

Reference to this string.







References [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#).

assign() [10/10]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::assign (
    size_type __n,
    _CharT __c ) [inline]
```

Set value to multiple characters.

Parameters

   	Length of the resulting string.
 	The character to use.

Returns

Reference to this string.

This function sets the value of this string to `__n` copies of character `__c`.

References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

at() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
reference std::basic_string< _CharT, _Traits, _Alloc >::at (
    size_type __n ) [inline]
```

Provides access to the data contained in the string.

Parameters

<code>_↔</code>	The index of the character to access.
<code>_n</code>	

Returns

Read/write reference to the character.

Exceptions

<code>std::out_of_range</code>	If <i>n</i> is an invalid index.
--------------------------------	----------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails. Success results in unsharing the string.
References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

at() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reference std::basic_string<_CharT, _Traits, _Alloc>::at (
    size_type __n ) const [inline]
```

Provides access to the data contained in the string.

Parameters

<code>_↔</code>	The index of the character to access.
<code>_n</code>	

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If <i>n</i> is an invalid index.
--------------------------------	----------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.
References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

back() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
reference std::basic_string<_CharT, _Traits, _Alloc>::back ( ) [inline]
```

Returns a read/write reference to the data at the last element of the string.

References [std::basic_string<_CharT, _Traits, _Alloc>::empty\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator\[\]\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

back() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

const_reference [std::basic_string<_CharT, _Traits, _Alloc>::back \(\)](#) const [inline], [noexcept]
 Returns a read-only (constant) reference to the data at the last element of the string.
 References [std::basic_string<_CharT, _Traits, _Alloc>::empty\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator\[\]\(\)](#),
 and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

begin() [1/2]

template<typename _CharT , typename _Traits , typename _Alloc >
 iterator [std::basic_string<_CharT, _Traits, _Alloc>::begin \(\)](#) [inline]
 Returns a read/write iterator that points to the first character in the string. Unshares the string.
 Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::crend\(\)](#), [std::regex_match\(\)](#), [std::regex_match\(\)](#),
[std::regex_replace\(\)](#), [std::regex_replace\(\)](#), [std::regex_search\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rend\(\)](#),
[std::basic_string<_CharT, _Traits, _Alloc>::rend\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

begin() [2/2]

template<typename _CharT , typename _Traits , typename _Alloc >
 const_iterator [std::basic_string<_CharT, _Traits, _Alloc>::begin \(\)](#) const [inline], [noexcept]
 Returns a read-only (constant) iterator that points to the first character in the string.

c_str()

template<typename _CharT , typename _Traits , typename _Alloc >
 const _CharT * [std::basic_string<_CharT, _Traits, _Alloc>::c_str \(\)](#) const [inline], [noexcept]
 Return const pointer to null-terminated contents.
 This is a handle to internal data. Do not modify or dire things may happen.
 Referenced by [std::collate<_CharT>::do_compare\(\)](#), [std::money_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#),
[std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::collate<_CharT>::do_transform\(\)](#),
[std::regex_replace\(\)](#), and [std::experimental::filesystem::v1::filesystem_error::what\(\)](#).

capacity()

template<typename _CharT , typename _Traits , typename _Alloc >
 size_type [std::basic_string<_CharT, _Traits, _Alloc>::capacity \(\)](#) const [inline], [noexcept]
 Returns the total number of characters that the string can hold before needing to allocate more memory.
 Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::push_back\(\)](#).

cbegin()

template<typename _CharT , typename _Traits , typename _Alloc >
 const_iterator [std::basic_string<_CharT, _Traits, _Alloc>::cbegin \(\)](#) const [inline], [noexcept]
 Returns a read-only (constant) iterator that points to the first character in the string.

cend()

template<typename _CharT , typename _Traits , typename _Alloc >
 const_iterator [std::basic_string<_CharT, _Traits, _Alloc>::cend \(\)](#) const [inline], [noexcept]
 Returns a read-only (constant) iterator that points one past the last character in the string.
 References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

clear()

template<typename _CharT , typename _Traits , typename _Alloc >
 void [std::basic_string<_CharT, _Traits, _Alloc>::clear \(\)](#) [inline], [noexcept]
 Erases the string, making it empty.
 References [std::basic_string<_CharT, _Traits, _Alloc>::get_allocator\(\)](#).

compare() [1/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string<_CharT, _Traits, _Alloc>::compare (
    const _CharT * __s ) const [noexcept]
```

Compare to a C string.

Parameters

<code>__s</code>	C string to compare against.
------------------	------------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__s`, 0 if their values are equivalent, or > 0 if this string is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and the length of a string constructed from `__s`. The function then compares the two strings by calling `traits::compare(data(),s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [std::min\(\)](#), and [std::size\(\)](#).

compare() [2/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv<_Tp, int > std::basic_string<_CharT, _Traits, _Alloc>::compare (
    const _Tp & __svt ) const [inline], [noexcept]
```

Compare to a string_view.

Parameters

<code>__svt</code>	An object convertible to <code>string_view</code> to compare against.
--------------------	---

Returns

Integer < 0, 0, or > 0.

References [std::min\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

compare() [3/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string<_CharT, _Traits, _Alloc>::compare (
    const basic_string<_CharT, _Traits, _Alloc> & __str ) const [inline]
```

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::min()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

compare() [4/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, int > std::basic_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos,
    size_type __n,
    const _Tp & __svt ) const [inline], [noexcept]
```

Compare to a `string_view`.

Parameters

<code>__pos</code>	A position in the string to start comparing from.
<code>__n</code>	The number of characters to compare.
<code>__svt</code>	An object convertible to <code>string_view</code> to compare against.

Returns

Integer < 0, 0, or > 0.

compare() [5/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos,
    size_type __n,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) const
```

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::min()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

`compare()` [6/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string<_CharT, _Traits, _Alloc>::compare (
    size_type __pos,
    size_type __n1,
    const _CharT * __s ) const
```

Compare substring to a C string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	C string to compare against.

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `pos`. Returns an integer < 0 if the substring is ordered before `__s`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and the length of a string constructed from `__s`. The function then compares the two string by calling `traits::compare(substring.data(), __s, rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References `std::min()`.

`compare()` [7/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic_string<_CharT, _Traits, _Alloc>::compare (
    size_type __pos,
    size_type __n1,
    const _CharT * __s,
    size_type __n2 ) const
```

Compare substring against a character array.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	character array to compare against.
<code>__n2</code>	Number of characters of <code>s</code> .

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `__pos`. Form a string from the first `__n2` characters of `__s`. Returns an integer < 0 if this substring is ordered before the string from `__s`, 0 if their values are equivalent, or > 0 if this substring is ordered after the string from `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__n2`. The function then compares the two strings by calling `traits::compare(substring.data(), __s, rlen)`.

::compare(substring.data(),s,rlen). If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

NB: s must have at least n2 characters, '\0' has no special meaning.

References [std::min\(\)](#).

compare() [8/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, int > std::basic\_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos1,
    size_type __n1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n2 = npos ) const [inline], [noexcept]
```

Compare to a string_view.

Parameters

<code>__pos1</code>	A position in the string to start comparing from.
<code>__n1</code>	The number of characters to compare.
<code>__svt</code>	An object convertible to string_view to compare against.
<code>__pos2</code>	A position in the string_view to start comparing from.
<code>__n2</code>	The number of characters to compare.

Returns

Integer < 0, 0, or > 0.

compare() [9/9]

```
template<typename _CharT , typename _Traits , typename _Alloc >
int std::basic\_string< _CharT, _Traits, _Alloc >::compare (
    size_type __pos1,
    size_type __n1,
    const basic\_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos2,
    size_type __n2 = npos ) const
```

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of str.
<code>__n2</code>	Number of characters in substring of str.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(),str.substr(pos2,n2).data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), and [std::min\(\)](#).

copy()

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc>::size_type basic_string<_CharT, _Traits, _Alloc>::copy (
    _CharT * __s,
    size_type __n,
    size_type __pos = 0 ) const
```

Copy substring into C string.

Parameters

<code>__s</code>	C string to copy value into.
<code>__n</code>	Number of characters to copy.
<code>__pos</code>	Index of first character to copy.

Returns

Number of characters actually copied

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
--------------------------------	-------------------------------------

Copies up to `__n` characters starting at `__pos` into the C string `__s`. If `__pos` is greater than `size()`, `out_of_range` is thrown.

crbegin()

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reverse_iterator std::basic_string<_CharT, _Traits, _Alloc>::crbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

References [std::basic_string<_CharT, _Traits, _Alloc>::end\(\)](#).

crend()

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reverse_iterator std::basic_string<_CharT, _Traits, _Alloc>::crend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

References [std::basic_string<_CharT, _Traits, _Alloc>::begin\(\)](#).

data() [1/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
const _CharT * std::basic_string<_CharT, _Traits, _Alloc>::data ( ) const [inline], [noexcept]
```

Return const pointer to contents.

This is a pointer to internal data. It is undefined to modify the contents through the returned pointer. To get a pointer that allows modifying the contents use `&str[0]` instead, (or in C++17 the non-const `str.data()` overload).

Referenced by [std::basic_regex<_Ch_type, _Rx_traits>::assign\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::compare\(\)](#), [std::collate<_CharT>::do_compare\(\)](#), [std::collate<_CharT>::do_transform\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), [std::match_results<_Bi_iter, _Alloc>::format\(\)](#), [std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::from_bytes\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::operator __sv_type\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rfind\(\)](#), [std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::to_bytes\(\)](#), [std::regex_traits<_Ch_type>::transform\(\)](#), and [std::filesystem::filesystem_error::what\(\)](#).

data() [2/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
_CharT * std::basic_string<_CharT, _Traits, _Alloc>::data ( ) [inline], [noexcept]
```

Return non-const pointer to contents.

This is a pointer to the character sequence held by the string. Modifying the characters in the sequence is allowed.

empty()

```
template<typename _CharT, typename _Traits, typename _Alloc>
bool std::basic_string<_CharT, _Traits, _Alloc>::empty ( ) const [inline], [noexcept]
```

Returns true if the string is empty. Equivalent to `*this == ""`.

References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::back\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::back\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::front\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::front\(\)](#), [std::tr2::operator>>\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::pop_back\(\)](#).

end() [1/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
iterator std::basic_string<_CharT, _Traits, _Alloc>::end ( ) [inline]
```

Returns a read/write iterator that points one past the last character in the string. Unshares the string.

References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::crbegin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::crbegin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rbegin\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::rbegin\(\)](#), [std::regex_match\(\)](#), [std::regex_match\(\)](#), [std::regex_replace\(\)](#), [std::regex_replace\(\)](#), and [std::regex_search\(\)](#).

end() [2/2]

```
template<typename _CharT, typename _Traits, typename _Alloc>
const_iterator std::basic_string<_CharT, _Traits, _Alloc>::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last character in the string.

References [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

erase() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::iterator std::basic_string< _CharT, _Traits, _Alloc >↵
::erase (
    iterator __first,
    iterator __last )
```

Remove a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to remove.
<code>__last</code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range [first,last) from this string. The value of the string doesn't change if an error is thrown.

erase() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
iterator std::basic_string< _CharT, _Traits, _Alloc >::erase (
    iterator __position ) [inline]
```

Remove one character.

Parameters

<code>__position</code>	Iterator referencing the character to remove.
-------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

erase() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::erase (
    size_type __pos = 0,
    size_type __n = npos ) [inline]
```

Remove characters.

Parameters

<code>__pos</code>	Index of first character to remove (default 0).
<code>__n</code>	Number of characters to remove (default remainder).

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is beyond the end of this string.
--------------------------------	---

Removes *__n* characters from this string starting at *__pos*. The length of the string is reduced by *__n*. If there are $< _n$ characters to remove, the remainder of the string is truncated. If *__p* is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Referenced by [std::getline\(\)](#), [std::operator>>\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::pop_back\(\)](#).

find() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::find (
    _CharT __c,
    size_type __pos = 0 ) const [noexcept]
```

Find position of a character.

Parameters

<i>__c</i>	Character to locate.
<i>__pos</i>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from *__pos*, searches forward for *__c* within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::find (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find position of a C substring.

Parameters

<i>__s</i>	C string to locate.
<i>__pos</i>	Index of character to search from.
<i>__n</i>	Number of characters from <i>s</i> to search for.

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References `std::data()`, and `std::size()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::find()`, `std::basic_string<_CharT, _Traits, _Alloc>::find()`, `std::basic_string<_CharT, _Traits, _Alloc>::find()`, and `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`.

find() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::find (
    const _CharT * __s,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References `std::basic_string<_CharT, _Traits, _Alloc>::find()`.

find() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv<_Tp, size_type > std::basic_string<_CharT, _Traits, _Alloc>::find (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a `string_view`.

Parameters

<code>__svt</code>	The object convertible to <code>string_view</code> to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

References `std::basic_string<_CharT, _Traits, _Alloc>::find()`.

find() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::find (
```



```
const basic_string< _CharT, _Traits, _Alloc > & __str,
size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::find\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

find_first_not_of() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::find_first_not_of (
    _CharT __c,
    size_type __pos = 0 ) const [noexcept]
```

Find position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_first_not_of() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::find_first_not_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>__s</code> to consider.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_first_not_of() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string<\_CharT, \_Traits, \_Alloc>::find\_first\_not\_of (
    const _CharT * __s,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#).

find_first_not_of() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv<_Tp, size_type > std::basic\_string<\_CharT, \_Traits, \_Alloc>::find\_first\_not\_of (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character not in a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#).

find_first_not_of() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string<\_CharT, \_Traits, \_Alloc>::find\_first\_not\_of (
    const basic\_string<\_CharT, \_Traits, \_Alloc> & __str,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of\(\)](#).

find_first_of() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::find_first_of (
    _CharT __c,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for the character `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `find(__c, __pos)`.

References [std::basic_string<_CharT, _Traits, _Alloc>::find\(\)](#).

find_first_of() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc>::size_type std::basic_string<_CharT, _Traits, _Alloc>::find_first_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find position of a character of C substring.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_first_of() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string< _CharT, _Traits, _Alloc >::find_first_of (
    const _CharT * __s,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character of C string.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::find_first_of\(\)](#).

find_first_of() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic\_string< _CharT, _Traits, _Alloc >::find_first_of (
    const _Tp & __svt,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character of a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

References [std::basic_string< _CharT, _Traits, _Alloc >::find_first_of\(\)](#).

find_first_of() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string< _CharT, _Traits, _Alloc >::find_first_of (
    const basic\_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = 0 ) const [inline], [noexcept]
```

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::find_first_of\(\)](#).

find_last_not_of() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc >::size_type std::basic_string<_CharT, _Traits, _Alloc
>::find_last_not_of (
    _CharT __c,
    size_type __pos = npos ) const [noexcept]
```

Find last position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_last_not_of() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string<_CharT, _Traits, _Alloc >::size_type std::basic_string<_CharT, _Traits, _Alloc
>::find_last_not_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find last position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to consider.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in the first `n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_last_not_of() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of (
    const _CharT * __s,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of\(\)](#).

find_last_not_of() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character not in a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

References [std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of\(\)](#).

find_last_not_of() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of (
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```


Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

Referenced by [std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of\(\)](#).

find_last_of() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_of (
    _CharT __c,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `rfind(__c, __pos)`.

References [std::basic_string< _CharT, _Traits, _Alloc >::rfind\(\)](#).

find_last_of() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::find_last_of (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find last position of a character of C substring.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

find_last_of() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_of (
    const _CharT * __s,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character of C string.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#).

find_last_of() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic_string< _CharT, _Traits, _Alloc >::find_last_of (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character of string.

Parameters

<code>__svt</code>	An object convertible to <code>string_view</code> containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

References [std::basic_string< _CharT, _Traits, _Alloc >::find_last_of\(\)](#).

find_last_of() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_of (
    const basic\_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::find_last_of\(\)](#).

front() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
reference std::basic\_string<\_CharT, \_Traits, \_Alloc>::front \( \) [inline]
```

Returns a read/write reference to the data at the first element of the string.

References [std::basic_string<_CharT, _Traits, _Alloc>::empty\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::operator\[\]\(\)](#).

front() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reference std::basic\_string<\_CharT, \_Traits, \_Alloc>::front \( \) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the string.

References [std::basic_string<_CharT, _Traits, _Alloc>::empty\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::operator\[\]\(\)](#).

get_allocator()

```
template<typename _CharT , typename _Traits , typename _Alloc >
allocator_type std::basic\_string<\_CharT, \_Traits, \_Alloc>::get\_allocator \( \) const [inline],
[noexcept]
```

Return copy of allocator used to construct this string.

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::~~basic_string\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::clear\(\)](#), [std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::from_bytes\(\)](#), [std::operator+\(\)](#), [std::operator+\(\)](#), and [std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::to_bytes\(\)](#).

insert() [1/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
iterator std::basic\_string<\_CharT, \_Traits, \_Alloc>::insert \(
    iterator __p,
    _CharT __c ) [inline]
```

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

insert() [2/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<class _InputIterator >
void std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    _InputIterator __beg,
    _InputIterator __end ) [inline]
```

Insert a range of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts characters in range `[__beg, __end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `std::basic_string<_CharT, _Traits, _Alloc>::replace()`.

insert() [3/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string<_CharT, _Traits, _Alloc>::insert (
    iterator __p,
    initializer_list<_CharT> __l ) [inline]
```

Insert an `initializer_list` of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [4/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::insert (
    iterator __p,
    size_type __n,
    _CharT __c ) [inline]
```

Insert multiple characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [5/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos,
    const _CharT * __s ) [inline]
```

Insert a C string.

Parameters

<code>__pos</code>	Position in string to insert at.
<code>__s</code>	The C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>pos</code> is beyond the end of this string.

Inserts the first n characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

insert() [6/11]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_string<_CharT, _Traits, _Alloc> & std::basic_string<_CharT, _Traits, _Alloc>::insert (
    size_type __pos,
    const _CharT * __s,
    size_type __n )
```

Insert a C substring.

Parameters

<code>__pos</code>	Position in string to insert at.
<code>__s</code>	The C string to insert.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts the first n characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

insert() [7/11]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, basic_string &> std::basic_string<_CharT, _Traits, _Alloc>::insert (
    size_type __pos,
    const _Tp & __svt ) [inline]
```

Insert a string_view.

Parameters

<code>__pos</code>	Position in string to insert at.
<code>__svt</code>	The object convertible to string_view to insert.

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::insert()`.

insert() [8/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos,
    size_type __n,
    _CharT __c ) [inline]
```

Insert multiple characters.

Parameters

<code>__pos</code>	Index in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts `__n` copies of character `__c` starting at index `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos > length()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

insert() [9/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n = npos ) [inline]
```

Insert a string_view.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__svt</code>	The object convertible to string_view to insert from.
<code>__pos2</code>	Position in string_view to insert from.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::replace()`.

insert() [10/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos1,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Insert value of a string.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References `std::basic_string<_CharT, _Traits, _Alloc>::insert()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

insert() [11/11]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::insert (
    size_type __pos1,
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos2,
    size_type __n = npos ) [inline]
```

Insert a substring.

Parameters

<code>__pos1</code>	Position in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>pos1 > size()</code> or <code>__pos2 > str.size()</code> .

Starting at `pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#).

length()

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::length ( ) const [inline], [noexcept]
```

Returns the number of characters in the string, not including any null-termination.

Referenced by [std::collate<_CharT>::do_compare\(\)](#), [std::collate<_CharT>::do_transform\(\)](#), [std::regex_traits<_Ch_type>::length\(\)](#), [std::regex_replace\(\)](#), and [std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::to_bytes\(\)](#).

max_size()

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::max_size ( ) const [inline], [noexcept]
```

Returns the size() of the largest possible string.

Referenced by [std::getline\(\)](#), and [std::operator>>\(\)](#).

operator __sv_type()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_string<_CharT, _Traits, _Alloc>::operator __sv_type ( ) const [inline], [noexcept]
```

Convert to a string_view.

Returns

A string_view.

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

operator+=() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::operator+= (
    _CharT __c ) [inline]
```

Append a character.

Parameters

<code>__c</code>	The character to append.
------------------	--------------------------

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::push_back\(\)](#).

operator+=() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator+= (
    const _CharT * __s ) [inline]
```

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`.

operator+=() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string< _CharT, _Traits, _Alloc >::operator+= (
    const _Tp & __svt ) [inline]
```

Append a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to be appended.
--------------------	---

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`.

operator+=() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator+= (
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`.

operator+=() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator+= (
    initializer_list< _CharT > __l ) [inline]
```

Append an initializer_list of characters.

Parameters

↩	The initializer_list of characters to be appended.
↩	
↩	
↩	
/	

Returns

Reference to this string.

References [std::basic_string< _CharT, _Traits, _Alloc >::append\(\)](#).

operator=() [1/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator= (
    _CharT __c ) [inline]
```

Set value to string of length 1.

Parameters

↩	Source character.
__c	

Assigning to a character makes this string length 1 and (*this)[0] == c.

References [std::basic_string< _CharT, _Traits, _Alloc >::assign\(\)](#).

operator=() [2/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator= (
    basic_string< _CharT, _Traits, _Alloc > && __str ) [inline], [noexcept]
```

Move assign the value of *str* to this string.

Parameters

__str	Source string.
-------	----------------

The contents of *str* are moved into this string (without copying). *str* is a valid, but unspecified string.

References [std::basic_string< _CharT, _Traits, _Alloc >::swap\(\)](#).

operator=() [3/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::operator= (
    const _CharT * __s ) [inline]
```

Copy contents of *s* into this string.

Parameters

<code>_↔</code>	Source null-terminated string.
<code>_S</code>	

References [std::basic_string<_CharT, _Traits, _Alloc >::assign\(\)](#).

operator=() [4/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic\_string & > std::basic\_string< _CharT, _Traits, _Alloc >::operator= (
    const _Tp & __svt ) [inline]
```

Set value to string constructed from a string_view.

Parameters

<code>__svt</code>	An object convertible to string_view.
------------------------------------	---------------------------------------

References [std::basic_string<_CharT, _Traits, _Alloc >::assign\(\)](#).

operator=() [5/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic\_string & std::basic\_string< _CharT, _Traits, _Alloc >::operator= (
    const basic\_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Assign the value of *str* to this string.

Parameters

<code>__str</code>	Source string.
------------------------------------	----------------

References [std::basic_string<_CharT, _Traits, _Alloc >::assign\(\)](#).

operator=() [6/6]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic\_string & std::basic\_string< _CharT, _Traits, _Alloc >::operator= (
    initializer\_list< _CharT > __l ) [inline]
```

Set value to string constructed from initializer list.

Parameters

<code>↔</code>	std::initializer_list .
<code>_↔</code>	
<code>↔</code>	
<code>_↔</code>	
<code>/</code>	

References [std::basic_string<_CharT, _Traits, _Alloc >::assign\(\)](#).

operator[]() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
reference std::basic_string< _CharT, _Traits, _Alloc >::operator[] (
    size_type __pos ) [inline]
```

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read/write reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.) Unshares the string.

References [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

operator[]() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reference std::basic_string< _CharT, _Traits, _Alloc >::operator[] (
    size_type __pos ) const [inline], [noexcept]
```

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read-only (constant) reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

References [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

Referenced by [std::basic_string< _CharT, _Traits, _Alloc >::back\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::back\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::front\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::front\(\)](#).

pop_back()

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::pop_back ( ) [inline]
```

Remove the last character.

The string must be non-empty.

References [std::basic_string< _CharT, _Traits, _Alloc >::empty\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::erase\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

push_back()

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::push_back (
    _CharT __c ) [inline]
```

Append a single character.

Parameters

<code>_↔</code>	Character to append.
<code>_C</code>	

References `std::basic_string<_CharT, _Traits, _Alloc>::capacity()`, `std::basic_string<_CharT, _Traits, _Alloc>::reserve()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

Referenced by `std::collate<_CharT>::do_transform()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator+=()`, `std::tr2::operator>>()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`.

rbegin() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rbegin ( ) [inline]
```

Returns a read/write reverse iterator that points to the last character in the string. Iteration is done in reverse element order. Unshares the string.

References `std::basic_string<_CharT, _Traits, _Alloc>::end()`.

rbegin() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

References `std::basic_string<_CharT, _Traits, _Alloc>::end()`.

rend() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rend ( ) [inline]
```

Returns a read/write reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order. Unshares the string.

References `std::basic_string<_CharT, _Traits, _Alloc>::begin()`.

rend() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
const_reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

References `std::basic_string<_CharT, _Traits, _Alloc>::begin()`.

replace() [1/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, basic_string & > std::basic_string< _CharT, _Traits, _Alloc >::replace (
    const_iterator __i1,
    const_iterator __i2,
    const _Tp & __svt ) [inline]
```

Replace range of characters with string_view.

Parameters

<code>__i1</code>	An iterator referencing the start position to replace at.
<code>__i2</code>	An iterator referencing the end position for the replace.
<code>__svt</code>	The object convertible to <code>string_view</code> to insert from.

Returns

Reference to this string.

References `std::basic_string<_CharT, _Traits, _Alloc>::begin()`, and `std::basic_string<_CharT, _Traits, _Alloc>::replace()`.

replace() [2/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<class _InputIterator >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    _InputIterator __k1,
    _InputIterator __k2 ) [inline]
```

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, characters in the range `[__k1,__k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [3/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    const _CharT * __s ) [inline]
```

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [4/14]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    iterator __i1,
    iterator __i2,
    const _CharT * __s,
    size_type __n ) [inline]
```

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from <code>s</code> to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the first `__n` characters of `__s` are inserted. If the length of

result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.
References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [5/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the value of `__str` is inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

replace() [6/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    iterator __i1,
    iterator __i2,
    initializer_list< _CharT > __l ) [inline]
```

Replace range of characters with `initializer_list`.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, characters in the range `[__k1, __k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [7/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    iterator __i1,
    iterator __i2,
    size_type __n,
    _CharT __c ) [inline]
```

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [8/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv<_Tp, basic_string & > std::basic_string<_CharT, _Traits, _Alloc>::replace (
    size_type __pos,
    size_type __n,
    const _Tp & __svt ) [inline]
```

Replace range of characters with string_view.

Parameters

<code>__pos</code>	The position to replace at.
--------------------	-----------------------------

Parameters

<code>__n</code>	The number of characters to replace.
<code>__svt</code>	The object convertible to <code>string_view</code> to insert.

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [9/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    size_type __pos,
    size_type __n,
    const basic_string< _CharT, _Traits, _Alloc > & __str ) [inline]
```

Replace characters with value from another string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is beyond the end of this string.
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos+__n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::size\(\)](#).

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::insert\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [10/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    size_type __pos,
```

```
size_type __n1,
const _CharT * __s ) [inline]
```

Replace characters with value of a C string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos + __n1)` from this string. In place, the characters of `__s` are inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [11/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::replace
(
    size_type __pos,
    size_type __n1,
    const _CharT * __s,
    size_type __n2 )
```

Replace characters with value of a C substring.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.
<code>__n2</code>	Number of characters from <code>s</code> to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos1 > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos + __n1)` from this string. In place, the first `__n2` characters of `__s` are inserted, or all of `__s` if `__n2` is too large. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [12/14]

```
template<typename _CharT, typename _Traits, typename _Alloc>
basic_string & std::basic_string<_CharT, _Traits, _Alloc>::replace (
    size_type __pos,
    size_type __n1,
    size_type __n2,
    _CharT __c ) [inline]
```

Replace characters with multiple characters.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__n2</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, `__n2` copies of `__c` are inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

replace() [13/14]

```
template<typename _CharT, typename _Traits, typename _Alloc>
template<typename _Tp>
_If_sv<_Tp, basic_string &> std::basic_string<_CharT, _Traits, _Alloc>::replace (
    size_type __pos1,
    size_type __n1,
    const _Tp & __svt,
    size_type __pos2,
    size_type __n2 = npos ) [inline]
```

Replace range of characters with string_view.

Parameters

<code>__pos1</code>	The position to replace at.
<code>__n1</code>	The number of characters to replace.
<code>__svt</code>	The object convertible to string_view to insert from.
<code>__pos2</code>	The position in the string_view to insert from.

Parameters

<code>__n2</code>	The number of characters to insert.
-------------------	-------------------------------------

Returns

Reference to this string.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

replace() [14/14]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string & std::basic_string< _CharT, _Traits, _Alloc >::replace (
    size_type __pos1,
    size_type __n1,
    const basic_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos2,
    size_type __n2 = npos ) [inline]
```

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of str to use.
<code>__n2</code>	Number of characters from str to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos1, __pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::replace\(\)](#).

reserve() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::reserve
Equivalent to shrink_to_fit().
```

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::push_back\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::shrink_to_fit\(\)](#).

reserve() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::reserve (
    size_type __res_arg )
```

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<code>std::length_error</code>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

References [std::size\(\)](#).

Referenced by [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::tr2::operator>>\(\)](#), and [std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow\(\)](#).

resize() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::resize (
    size_type __n ) [inline]
```

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
------------------	---

This function will resize the string to the specified length. If the new size is smaller than the string's current size the string is truncated, otherwise the string is extended and new characters are default-constructed. For basic types such as `char`, this means setting them to 0.

References [std::basic_string< _CharT, _Traits, _Alloc >::resize\(\)](#).

resize() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::resize (
    size_type __n,
    _CharT __c )
```

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
------------------	---

Parameters

<code>__c</code>	Character to fill any new elements.
------------------	-------------------------------------

This function will resize the string to the specified number of characters. If the number is smaller than the string's current size the string is truncated, otherwise the string is extended and new elements are set to `__c`.

References [std::size\(\)](#).

Referenced by [std::money_get<_CharT, _InIter>::do_get\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::resize\(\)](#).

rfind() [1/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::rfind (
    _CharT __c,
    size_type __pos = npos ) const [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

References [std::size\(\)](#).

rfind() [2/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc
>::rfind (
    const _CharT * __s,
    size_type __pos,
    size_type __n ) const [noexcept]
```

Find last position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::min\(\)](#), and [std::size\(\)](#).

rfind() [3/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string< _CharT, _Traits, _Alloc >::rfind (
    const _CharT * __s,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to start search at (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References [std::basic_string< _CharT, _Traits, _Alloc >::rfind\(\)](#).

rfind() [4/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
template<typename _Tp >
_If_sv< _Tp, size_type > std::basic\_string< _CharT, _Traits, _Alloc >::rfind (
    const _Tp & __svt,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a string_view.

Parameters

<code>__svt</code>	The object convertible to string_view to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

References [std::basic_string< _CharT, _Traits, _Alloc >::rfind\(\)](#).

rfind() [5/5]

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic\_string< _CharT, _Traits, _Alloc >::rfind (
    const basic\_string< _CharT, _Traits, _Alloc > & __str,
    size_type __pos = npos ) const [inline], [noexcept]
```

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::find_last_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`, `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`, and `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`.

shrink_to_fit()

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string<_CharT, _Traits, _Alloc>::shrink_to_fit ( ) [inline], [noexcept]
```

A non-binding request to reduce capacity() to size().

References `std::basic_string<_CharT, _Traits, _Alloc>::reserve()`.

size()

```
template<typename _CharT , typename _Traits , typename _Alloc >
size_type std::basic_string<_CharT, _Traits, _Alloc>::size ( ) const [inline], [noexcept]
```

Returns the number of characters in the string, not including any null-termination.

Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_regex<_Ch_type, _Rx_traits>::assign()`, `std::basic_string<_CharT, _Traits, _Alloc>::assign()`, `std::basic_string<_CharT, _Traits, _Alloc>::at()`, `std::basic_string<_CharT, _Traits, _Alloc>::back()`, `std::basic_string<_CharT, _Traits, _Alloc>::back()`, `std::basic_string<_CharT, _Traits, _Alloc>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::empty()`, `std::basic_string<_CharT, _Traits, _Alloc>::end()`, `std::basic_string<_CharT, _Traits, _Alloc>::end()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_of()`, `std::match_results<_Bi_iter, _Alloc>::format()`, `std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::from_bytes()`, `std::basic_string<_CharT, _Traits, _Alloc>::insert()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator __sv_type()`, `std::operator+`, `std::operator+`, `std::tr2::operator>>()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator[]()`, `std::basic_string<_CharT, _Traits, _Alloc>::operator[]()`, `std::basic_string<_CharT, _Traits, _Alloc>::pop_back()`, `std::basic_string<_CharT, _Traits, _Alloc>::push_back()`, `std::basic_string<_CharT, _Traits, _Alloc>::replace()`, `std::basic_string<_CharT, _Traits, _Alloc>::replace()`, `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`, `std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::to_bytes()`, and `std::regex_traits<_Ch_type>::transform()`.

substr()

```
template<typename _CharT , typename _Traits , typename _Alloc >
basic_string std::basic_string<_CharT, _Traits, _Alloc>::substr (
    size_type __pos = 0,
    size_type __n = npos ) const [inline]
```

Get a substring.

Parameters

<code>__pos</code>	Index of first character (default 0).
<code>__n</code>	Number of characters in substring (default remainder).

Returns

The new string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
--------------------------------	-------------------------------------

Construct and return a new string using the `__n` characters starting at `__pos`. If the string is too short, use the remainder of the characters. If `__pos` is beyond the end of the string, `out_of_range` is thrown.

References [std::basic_string<_CharT, _Traits, _Alloc>::basic_string\(\)](#).

swap()

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_string< _CharT, _Traits, _Alloc >::swap (
    basic_string< _CharT, _Traits, _Alloc > & __s ) [noexcept]
```

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

Referenced by [std::basic_string<_CharT, _Traits, _Alloc>::assign\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::operator=\(\)](#).

6.259.4 Member Data Documentation**npos**

```
template<typename _CharT , typename _Traits , typename _Alloc >
const basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _↵
_Alloc >::npos [static]
```

Value returned by various member functions when they fail.

The documentation for this class was generated from the following files:

- [cow_string.h](#)
- [basic_string.tcc](#)

6.260 std::basic_string_view<_CharT, _Traits> Class Template Reference

```
#include <>>
```

Public Types

- using **const_iterator** = const value_type *
- using **const_pointer** = const value_type *
- using **const_reference** = const value_type &
- using **const_reverse_iterator** = [std::reverse_iterator](#)< const_iterator >
- using **difference_type** = ptrdiff_t
- using **iterator** = const_iterator
- using **pointer** = value_type *
- using **reference** = value_type &

- using **reverse_iterator** = [const_reverse_iterator](#)
- using **size_type** = size_t
- using **traits_type** = _Traits
- using **value_type** = _CharT

Public Member Functions

- template<contiguous_iterator _It, sized_sentinel_for< _It > _End>
requires same_as<iter_value_t<_It>, _CharT> && (!convertible_to<_End, size_type>)
constexpr **basic_string_view** (_It __first, _End __last) noexcept(noexcept(__last - __first))
- constexpr **basic_string_view** (const _CharT * __str) noexcept
- constexpr **basic_string_view** (const _CharT * __str, size_type __len) noexcept
- constexpr **basic_string_view** (const [basic_string_view](#) &) noexcept=default
- constexpr const_reference **at** (size_type __pos) const
- constexpr const_reference **back** () const noexcept
- constexpr const_iterator **begin** () const noexcept
- constexpr const_iterator **cbegin** () const noexcept
- constexpr const_iterator **cend** () const noexcept
- constexpr int **compare** ([basic_string_view](#) __str) const noexcept
- constexpr int **compare** (const _CharT * __str) const noexcept
- constexpr int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str) const
- constexpr int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str, size_type __pos2, size_type __n2) const
- constexpr int **compare** (size_type __pos1, size_type __n1, const _CharT * __str) const
- constexpr int **compare** (size_type __pos1, size_type __n1, const _CharT * __str, size_type __n2) const noexcept(false)
- constexpr size_type **copy** (_CharT * __str, size_type __n, size_type __pos=0) const
- constexpr [const_reverse_iterator](#) **crbegin** () const noexcept
- constexpr [const_reverse_iterator](#) **crend** () const noexcept
- constexpr const_pointer **data** () const noexcept
- constexpr bool **empty** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr bool **ends_with** (_CharT __x) const noexcept
- constexpr bool **ends_with** ([basic_string_view](#) __x) const noexcept
- constexpr bool **ends_with** (const _CharT * __x) const noexcept
- constexpr size_type **find** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find** (const _CharT * __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find_first_not_of** (const _CharT * __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find_first_of** (const _CharT * __str, size_type __pos=0) const noexcept
- constexpr size_type **find_last_not_of** (_CharT __c, size_type __pos=npow) const noexcept
- constexpr size_type **find_last_not_of** ([basic_string_view](#) __str, size_type __pos=npow) const noexcept
- constexpr size_type **find_last_not_of** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find_last_not_of** (const _CharT * __str, size_type __pos=npow) const noexcept

- constexpr size_type **find_last_of** (_CharT __c, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_of** (basic_string_view __str, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_of** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find_last_of** (const _CharT * __str, size_type __pos=npos) const noexcept
- constexpr const_reference **front** () const noexcept
- constexpr size_type **length** () const noexcept
- constexpr size_type **max_size** () const noexcept
- constexpr basic_string_view & **operator=** (const basic_string_view &) noexcept=default
- constexpr const_reference **operator[]** (size_type __pos) const noexcept
- constexpr const_reverse_iterator **rbegin** () const noexcept
- constexpr void **remove_prefix** (size_type __n) noexcept
- constexpr void **remove_suffix** (size_type __n) noexcept
- constexpr const_reverse_iterator **rend** () const noexcept
- constexpr size_type **rfind** (_CharT __c, size_type __pos=npos) const noexcept
- constexpr size_type **rfind** (basic_string_view __str, size_type __pos=npos) const noexcept
- constexpr size_type **rfind** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **rfind** (const _CharT * __str, size_type __pos=npos) const noexcept
- constexpr size_type **size** () const noexcept
- constexpr bool **starts_with** (_CharT __x) const noexcept
- constexpr bool **starts_with** (basic_string_view __x) const noexcept
- constexpr bool **starts_with** (const _CharT * __x) const noexcept
- constexpr basic_string_view **substr** (size_type __pos=0, size_type __n=npos) const noexcept(false)
- constexpr void **swap** (basic_string_view & __sv) noexcept

Static Public Attributes

- static constexpr size_type **npos**

6.260.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class std::basic_string_view<_CharT, _Traits >
```

A non-owning reference to a string.

Template Parameters

<code>_CharT</code>	Type of character
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

A `basic_string_view` looks like this:

```
_CharT*    _M_str
size_t     _M_len
```

The documentation for this class was generated from the following files:

- [string_view](#)
- [bits/string_view.tcc](#)

6.261 std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits > Class Template Reference

```
#include <>>
```

Public Types

- using **const_iterator** = const _CharT *
- using **const_pointer** = const _CharT *
- using **const_reference** = const _CharT &
- using **const_reverse_iterator** = [std::reverse_iterator](#)< const_iterator >
- using **difference_type** = ptrdiff_t
- using **iterator** = const_iterator
- using **pointer** = _CharT *
- using **reference** = _CharT &
- using **reverse_iterator** = [const_reverse_iterator](#)
- using **size_type** = size_t
- using **traits_type** = _Traits
- using **value_type** = _CharT

Public Member Functions

- constexpr **basic_string_view** (const _CharT * __str)
- constexpr **basic_string_view** (const _CharT * __str, size_type __len)
- template<typename _Allocator >
 basic_string_view (const [basic_string](#)< _CharT, _Traits, _Allocator > & __str) noexcept
- constexpr **basic_string_view** (const [basic_string_view](#) &) noexcept=default
- constexpr const _CharT & **at** (size_type __pos) const
- constexpr const _CharT & **back** () const
- constexpr const_iterator **begin** () const noexcept
- constexpr const_iterator **cbegin** () const noexcept
- constexpr const_iterator **cend** () const noexcept
- constexpr int **compare** ([basic_string_view](#) __str) const noexcept
- constexpr int **compare** (const _CharT * __str) const noexcept
- constexpr int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str) const
- constexpr int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str, size_type __pos2, size_type __n2) const
- constexpr int **compare** (size_type __pos1, size_type __n1, const _CharT * __str) const
- constexpr int **compare** (size_type __pos1, size_type __n1, const _CharT * __str, size_type __n2) const
- size_type **copy** (_CharT * __str, size_type __n, size_type __pos=0) const
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- constexpr const _CharT * **data** () const noexcept
- constexpr bool **empty** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr size_type **find** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find** (const _CharT * __str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **find** (const _CharT * __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_not_of** (const _CharT * __str, size_type __pos, size_type __n) const
- constexpr size_type **find_first_not_of** (const _CharT * __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** (_CharT __c, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- constexpr size_type **find_first_of** (const _CharT * __str, size_type __pos, size_type __n) const
- constexpr size_type **find_first_of** (const _CharT * __str, size_type __pos=0) const noexcept

- constexpr size_type **find_last_not_of** (_CharT __c, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_not_of** (basic_string_view __str, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_not_of** (const _CharT *__str, size_type __pos, size_type __n) const
- constexpr size_type **find_last_not_of** (const _CharT *__str, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_of** (_CharT __c, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_of** (basic_string_view __str, size_type __pos=npos) const noexcept
- constexpr size_type **find_last_of** (const _CharT *__str, size_type __pos, size_type __n) const
- constexpr size_type **find_last_of** (const _CharT *__str, size_type __pos=npos) const noexcept
- constexpr const _CharT & **front** () const
- constexpr size_type **length** () const noexcept
- constexpr size_type **max_size** () const noexcept
- template<typename _Allocator >
operator basic_string< _CharT, _Traits, _Allocator > () const
- basic_string_view & **operator=** (const basic_string_view &) noexcept=default
- constexpr const _CharT & **operator[]** (size_type __pos) const
- const_reverse_iterator **rbegin** () const noexcept
- constexpr void **remove_prefix** (size_type __n)
- constexpr void **remove_suffix** (size_type __n)
- const_reverse_iterator **rend** () const noexcept
- constexpr size_type **rfind** (_CharT __c, size_type __pos=npos) const noexcept
- constexpr size_type **rfind** (basic_string_view __str, size_type __pos=npos) const noexcept
- constexpr size_type **rfind** (const _CharT *__str, size_type __pos, size_type __n) const noexcept
- constexpr size_type **rfind** (const _CharT *__str, size_type __pos=npos) const noexcept
- constexpr size_type **size** () const noexcept
- constexpr basic_string_view **substr** (size_type __pos=0, size_type __n=npos) const
- constexpr void **swap** (basic_string_view & __sv) noexcept
- template<typename _Allocator = std::allocator<_CharT>>
basic_string< _CharT, _Traits, _Allocator > **to_string** (const _Allocator & __alloc=_Allocator()) const

Static Public Attributes

- static constexpr size_type **npos**

6.261.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >
```

A non-owning reference to a string.

Template Parameters

<code>_CharT</code>	Type of character
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

A `basic_string_view` looks like this:

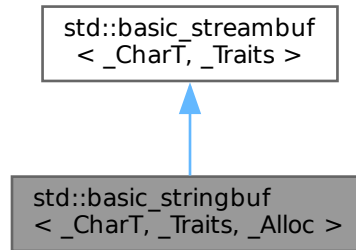
```
_CharT*      _M_str
size_t       _M_len
```

The documentation for this class was generated from the following files:

- [experimental/string_view](#)
- [experimental/bits/string_view.tcc](#)

6.262 std::basic_stringbuf< _CharT, _Traits, _Alloc > Class Template Reference

Inheritance diagram for std::basic_stringbuf< _CharT, _Traits, _Alloc >:



Public Types

- typedef `__string_type::size_type` **__size_type**
- typedef `basic_streambuf< char_type, traits_type >` **__streambuf_type**
- typedef `basic_string< char_type, _Traits, _Alloc >` **__string_type**
- typedef `_Alloc` **allocator_type**
- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `_Traits` **traits_type**

Public Member Functions

- `basic_stringbuf ()`
- `basic_stringbuf (basic_stringbuf && __rhs)`
- `basic_stringbuf (const __string_type & __str, ios_base::openmode __mode=ios_base::in|ios_base::out)`
- `basic_stringbuf (const basic_stringbuf &)=delete`
- `basic_stringbuf (ios_base::openmode __mode)`
- `locale getloc () const`
- `streamsize in_avail ()`
- `basic_stringbuf & operator= (basic_stringbuf && __rhs)`
- `basic_stringbuf & operator= (const basic_stringbuf &)=delete`
- `locale pubimbue (const locale & __loc)`
- `int_type sbumpc ()`
- `int_type sgetc ()`
- `streamsize sgetn (char_type * __s, streamsize __n)`
- `int_type snextc ()`
- `int_type sputbackc (char_type __c)`
- `int_type sputc (char_type __c)`
- `streamsize sputn (const char_type * __s, streamsize __n)`
- `__string_type str () const`
- `void str (const __string_type & __s)`

- int_type [sungetc](#) ()
- void [swap](#) ([basic_stringbuf](#) &__rhs) noexcept([_Noexcept_swap::value](#))
- [basic_streambuf](#) * [pubsetbuf](#) (char_type * __s, [streamsize](#) __n)
- pos_type [pubseekoff](#) (off_type __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- pos_type [pubseekpos](#) (pos_type __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- int [pubsync](#) ()

Protected Member Functions

- void [__safe_gbump](#) ([streamsize](#) __n)
- void [__safe_pbump](#) ([streamsize](#) __n)
- void [_M_pbump](#) (char_type * __pbeg, char_type * __pend, off_type __off)
- void [_M_stringbuf_init](#) ([ios_base::openmode](#) __mode)
- void [_M_sync](#) (char_type * __base, __size_type __i, __size_type __o)
- void [_M_update_egptr](#) ()
- void [gbump](#) (int __n)
- virtual void [imbue](#) (const [locale](#) &__loc)
- virtual int_type [overflow](#) (int_type __c=[traits_type::eof\(\)](#))
- virtual int_type [pbackfail](#) (int_type __c=[traits_type::eof\(\)](#))
- void [pbump](#) (int __n)
- virtual pos_type [seekoff](#) (off_type __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- virtual pos_type [seekpos](#) (pos_type __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- virtual [__streambuf_type](#) * [setbuf](#) (char_type * __s, [streamsize](#) __n)
- void [setg](#) (char_type * __gbeg, char_type * __gnext, char_type * __gend)
- void [setp](#) (char_type * __pbeg, char_type * __pend)
- virtual [streamsize](#) [showmanyc](#) ()
- void [swap](#) ([basic_streambuf](#) &__sb)
- virtual int [sync](#) ()
- virtual int_type [uflow](#) ()
- virtual int_type [underflow](#) ()
- virtual [streamsize](#) [xsgetn](#) (char_type * __s, [streamsize](#) __n)
- virtual [streamsize](#) [xspn](#) (const char_type * __s, [streamsize](#) __n)
- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const
- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- [locale](#) [_M_buf_locale](#)
- char_type * [_M_in_beg](#)
- char_type * [_M_in_cur](#)
- char_type * [_M_in_end](#)
- [ios_base::openmode](#) [_M_mode](#)
- char_type * [_M_out_beg](#)
- char_type * [_M_out_cur](#)
- char_type * [_M_out_end](#)
- [__string_type](#) [_M_string](#)

6.262.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_stringbuf< _CharT, _Traits, _Alloc >
```

The actual work of input and output (for std::string).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class associates either or both of its input and output sequences with a sequence of characters, which can be initialized from, or made available as, a `std::basic_string`. (Paraphrased from [27.7.1]/1.)

For this class, open modes (of type `ios_base::openmode`) have `in` set if the input sequence can be read, and `out` set if the output sequence can be written.

6.262.2 Constructor & Destructor Documentation

basic_stringbuf() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringbuf< _CharT, _Traits, _Alloc >::basic_stringbuf ( ) [inline]
```

Starts with an empty string buffer.

The default constructor initializes the parent class using its own default ctor.

basic_stringbuf() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringbuf< _CharT, _Traits, _Alloc >::basic_stringbuf (
    ios_base::openmode __mode ) [inline], [explicit]
```

Starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

The default constructor initializes the parent class using its own default ctor.

basic_stringbuf() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringbuf< _CharT, _Traits, _Alloc >::basic_stringbuf (
    const __string_type & __str,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [explicit]
```

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__mode</code>	Whether the buffer can read, or write, or both.

This constructor initializes the parent class using its own default ctor.

6.262.3 Member Function Documentation

eback()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::egptr ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow\(\)](#).

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- ep_ptr() returns the end pointer for the output sequence

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic\_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
Locale access.
```

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
Access to the get area.
```

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow()`.

imbue()

```
template<typename _CharT , typename _Traits >
virtual void std::basic_streambuf< _CharT, _Traits >::imbue (
    const locale & __loc ) [inline], [protected], [virtual], [inherited]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >`, `std::basic_filebuf< _CharT, std::char_traits< _CharT > >`, and `std::basic_filebuf< char_type, traits_type >`.

in_avail()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::in_avail ( ) [inline], [inherited]
Looking ahead into the stream.
```

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

overflow()

```
template<class _CharT , class _Traits , class _Alloc >
basic_stringbuf< _CharT, _Traits, _Alloc >::int_type std::basic_stringbuf< _CharT, _Traits, _↵
_Alloc >::overflow (
    int_type __c = traits_type::eof() ) [protected], [virtual]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>_↵</code>	An additional character to consume.
<code>_C</code>	

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::basic_string< _CharT, _Traits, _Alloc >::assign()`, `std::ios_base::in`, `std::max()`, `std::min()`, `std::ios_base::out`, `std::basic_string< _CharT, _Traits, _Alloc >::push_back()`, and `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`.

pbackfail()

```
template<class _CharT , class _Traits , class _Alloc >
basic_stringbuf< _CharT, _Traits, _Alloc >::int_type std::basic_stringbuf< _CharT, _Traits, _↵
_Alloc >::pbackfail (
    int_type __c = traits_type::eof() ) [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>_↵</code>	The character to be inserted back into the sequence.
<code>_C</code>	

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

References [std::ios_base::out](#).

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline], [inherited]
```


Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekpos function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of pubfoo dispatch to the protected derived foo member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual sync function.

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync\(\)](#), and [std::basic_istream< _CharT, _Traits >::sync\(\)](#).

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns [uflow\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< char >::ignore\(\)](#), [std::istreambuf_iterator< _CharT, _Traits >::operator++\(\)](#), and [std::basic_istream< char >::seekg\(\)](#).

seekoff()

```
template<class _CharT , class _Traits , class _Alloc >
basic_stringbuf< _CharT, _Traits, _Alloc >::pos_type std::basic_stringbuf< _CharT, _Traits, _↵
_Alloc >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

References [std::ios_base::cur](#), [std::ios_base::end](#), [std::ios_base::in](#), and [std::ios_base::out](#).

seekpos()

```
template<class _CharT , class _Traits , class _Alloc >
basic_stringbuf< _CharT, _Traits, _Alloc >::pos_type std::basic_stringbuf< _CharT, _Traits, _↵
_Alloc >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

References [std::ios_base::in](#), and [std::ios_base::out](#).

setbuf()

```
template<typename _CharT , typename _Traits , typename _Alloc >
virtual __streambuf_type * std::basic_stringbuf< _CharT, _Traits, _Alloc >::setbuf (
    char_type * __s,
    streamsize __n ) [inline], [protected], [virtual]
```

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no buffer has already been created, and both `__s` and `__n` are non-zero, then `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

setg()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::wbuffer_convert()`.

setp()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setp (
    char_type * __pbeg,
    char_type * __pend ) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::wbuffer_convert\(\)](#).

sgetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< _CharT, _Traits >::getline\(\)](#), [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::istreambuf_iterator< _CharT, _Traits >::operator++\(\)](#), and [std::basic_istream< char >::seekg\(\)](#).

sgetn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::sgetn (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
template<typename _CharT , typename _Traits , typename _Alloc >
virtual streamsize std::basic_stringbuf< _CharT, _Traits, _Alloc >::showmanyc ( ) [inline],
[protected], [virtual]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

snextc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::snextc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::seekg()`, and `std::basic_istream<char>::unget()`.

sputbackc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
    char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

$_c$	A character to output.
-------	------------------------

Returns

$_c$, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores $_c$ in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::ostreambuf_iterator< _CharT, _Traits >::operator=()`, and `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::overflow()`.

sputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::sputn (
    const char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

$_s$	A buffer read area.
$_n$	A count.

One of two public output functions.

Returns `xsputn(__s,__n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

str() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__string_type std::basic_stringbuf< _CharT, _Traits, _Alloc >::str ( ) const [inline]
```

Copying out the string buffer.

Returns

A copy of one of the underlying sequences.

If the buffer is only created in input mode, the underlying character sequence is equal to the input sequence; otherwise, it is equal to the output sequence. [27.7.1.2]/1

str() [2/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
void std::basic_stringbuf< _CharT, _Traits, _Alloc >::str (
    const __string_type & __s ) [inline]
```

Setting a new buffer.

Parameters

<code>_↔</code>	The string to use as a new sequence.
<code>_s</code>	

Deallocates any previous stored sequence, then copies *s* to use as a new one.

sungetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sungetc ( ) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Referenced by `std::basic_istream< _CharT, _Traits >::unget()`.

sync()

```
template<typename _CharT , typename _Traits >
virtual int std::basic_streambuf< _CharT, _Traits >::sync ( ) [inline], [protected], [virtual],
[inherited]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented in `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >`, `std::basic_filebuf< _CharT, std::c_encoding_char_traits< _CharT > >`, and `std::basic_filebuf< char_type, traits_type >`.

uflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual],
[inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`.

underflow()

```
template<class _CharT , class _Traits , class _Alloc >
basic_stringbuf< _CharT, _Traits, _Alloc >::int_type std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow [protected], [virtual]
Fetches more data from the controlled sequence.
```

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

References `std::ios_base::in`.

xsggetn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::xsggetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >`, `std::basic_filebuf< _CharT, std::char_traits< _CharT > >`, and `std::basic_filebuf< char_type, traits_type >`.

References `std::min()`.

xsputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::xsputn (
    const char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```


Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >](#), [std::basic_filebuf< _CharT, std::char_traits< _CharT > >](#), and [std::basic_filebuf< char_type, traits_type >](#).

References [std::min\(\)](#).

6.262.4 Member Data Documentation

`_M_buf_locale`

```
template<typename _CharT , typename _Traits >
```

```
locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale [protected], [inherited]
```

Current locale setting.

Referenced by [std::basic_filebuf< _CharT, _Traits >::basic_filebuf\(\)](#).

`_M_in_beg`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg [protected], [inherited]
```

Start of get area.

`_M_in_cur`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur [protected], [inherited]
```

Current read area.

`_M_in_end`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end [protected], [inherited]
```

End of get area.

`_M_mode`

```
template<typename _CharT , typename _Traits , typename _Alloc >
```

```
ios_base::openmode std::basic_stringbuf< _CharT, _Traits, _Alloc >::_M_mode [protected]
```

Place to stash in || out || in | out settings for current stringbuf.

`_M_out_beg`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg [protected], [inherited]
```

Start of put area.

_M_out_cur

```
template<typename _CharT , typename _Traits >
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_out_cur [protected], [inherited]
Current put area.
```

_M_out_end

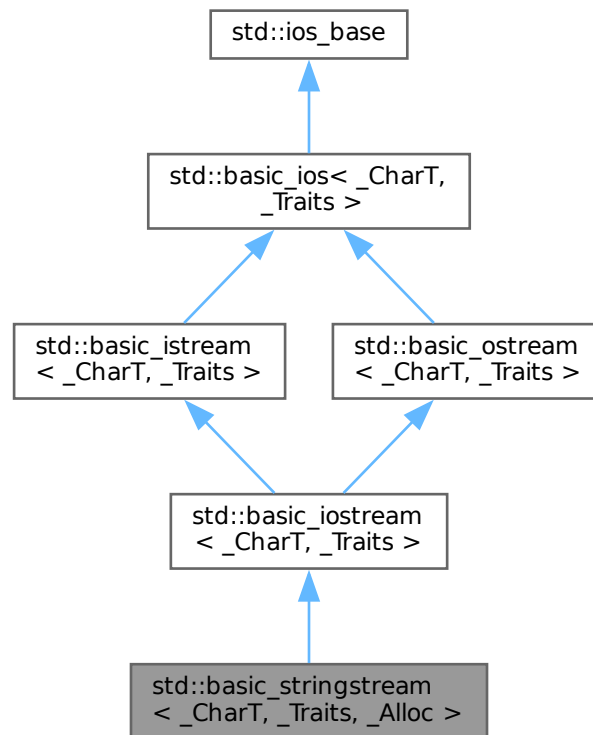
```
template<typename _CharT , typename _Traits >
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_out_end [protected], [inherited]
End of put area.
```

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)
- [sstream.tcc](#)

6.263 std::basic_stringstream< _CharT, _Traits, _Alloc > Class Template Reference

Inheritance diagram for `std::basic_stringstream< _CharT, _Traits, _Alloc >`:

**Public Types**

- typedef [ctype](#)< _CharT > `__ctype_type`

- typedef [ctype](#)<_CharT> **__ctype_type**
 - typedef [basic_ios](#)<_CharT, _Traits> **__ios_type**
 - typedef [basic_ios](#)<_CharT, _Traits> **__ios_type**
 - typedef [basic_istream](#)<char_type, traits_type> **__istream_type**
 - typedef [basic_istream](#)<_CharT, _Traits> **__istream_type**
 - typedef [num_get](#)<_CharT, [istreambuf_iterator](#)<_CharT, _Traits>> **__num_get_type**
 - typedef [num_put](#)<_CharT, [ostreambuf_iterator](#)<_CharT, _Traits>> **__num_put_type**
 - typedef [basic_ostream](#)<_CharT, _Traits> **__ostream_type**
 - typedef [basic_streambuf](#)<_CharT, _Traits> **__streambuf_type**
 - typedef [basic_streambuf](#)<_CharT, _Traits> **__streambuf_type**
 - typedef [basic_string](#)<_CharT, _Traits, _Alloc> **__string_type**
 - typedef [basic_stringbuf](#)<_CharT, _Traits, _Alloc> **__stringbuf_type**
 - typedef [_Alloc](#) **allocator_type**
 - typedef [_CharT](#) **char_type**
 - enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
 - typedef [_ios_Fmtflags](#) **fmtflags**
 - typedef traits_type::int_type **int_type**
 - typedef [_ios_iostate](#) **iostate**
 - typedef traits_type::off_type **off_type**
 - typedef [_ios_Openmode](#) **openmode**
 - typedef traits_type::pos_type **pos_type**
 - typedef [_ios_Seekdir](#) **seekdir**
 - typedef [_Traits](#) **traits_type**
-
- typedef [num_put](#)<_CharT, [ostreambuf_iterator](#)<_CharT, _Traits>> **__num_put_type**

Public Member Functions

- [basic_stringstream](#) ()
- **basic_stringstream** ([basic_stringstream](#) &&__rhs)
- [basic_stringstream](#) (const [__string_type](#) &__str, [ios_base::openmode](#) __m=[ios_base::out](#)|[ios_base::in](#))
- **basic_stringstream** (const [basic_stringstream](#) &)=delete
- [basic_stringstream](#) ([ios_base::openmode](#) __m)
- [~basic_stringstream](#) ()
- template<typename _ValueT>
[basic_istream](#)<_CharT, _Traits> & **_M_extract** (_ValueT &__v)
- const [locale](#) & **_M_getloc** () const
- template<typename _ValueT>
[basic_ostream](#)<_CharT, _Traits> & **_M_insert** (_ValueT __v)
- void **_M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & **copyfmt** (const [basic_ios](#) &__rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)
- [fmtflags](#) **flags** () const

- `fmtflags flags (fmtflags __fmtfl)`
 - `__ostream_type & flush ()`
 - `streamsize gcount () const`
 - `basic_istream< char > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `basic_istream< wchar_t > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `locale getloc () const`
 - `bool good () const`
 - `basic_istream< char > & ignore (streamsize __n)`
 - `basic_istream< wchar_t > & ignore (streamsize __n)`
 - `basic_istream< char > & ignore (streamsize __n, int_type __delim)`
 - `basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
 - `locale imbue (const locale & __loc)`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __dfault) const`
 - `__ostream_type & operator<< (__streambuf_type *__sb)`
 - `__ostream_type & operator<< (const void *__p)`
 - `__ostream_type & operator<< (nullptr_t)`
 - `basic_stringstream & operator= (basic_stringstream && __rhs)`
 - `basic_stringstream & operator= (const basic_stringstream &)=delete`
 - `__istream_type & operator>> (__streambuf_type *__sb)`
 - `__istream_type & operator>> (void *& __p)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `__stringbuf_type * rdbuf () const`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > *__sb)`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `__ostream_type & seekp (off_type, ios_base::seekdir)`
 - `__ostream_type & seekp (pos_type)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `__string_type str () const`
 - `void str (const __string_type & __s)`
 - `void swap (basic_stringstream & __rhs)`
 - `pos_type tellp ()`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > *__tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`
- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state (without causing an `ios_base::failure` to be thrown) and the original exception will be rethrown if `badbit` is set in the exceptions mask.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type *__s, streamsize __n)`
- `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
- `__istream_type & get (__streambuf_type &__sb)`
- `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type *__s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type *__s, streamsize __n)`
- `streamsize readsome (char_type *__s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`

- `operator bool () const`
- `bool operator! () const`
- `__ostream_type & operator<< (__ostream_type &(__pf)(__ostream_type &))`
- `__ostream_type & operator<< (__ios_type &(__pf)(__ios_type &))`
- `__ostream_type & operator<< (ios_base &(__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`
- `__ostream_type & operator<< (short __n)`
- `__ostream_type & operator<< (unsigned short __n)`
- `__ostream_type & operator<< (int __n)`
- `__ostream_type & operator<< (unsigned int __n)`
- `__ostream_type & operator<< (long long __n)`
- `__ostream_type & operator<< (unsigned long long __n)`
- `__ostream_type & operator<< (double __f)`
- `__ostream_type & operator<< (float __f)`
- `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `__ostream_type & write (const char_type *__s, streamsize __n)`

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const `openmode` `__noreplace`
- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`

- static const [fmtflags](#) basefield
- static const [seekdir](#) beg
- static const [openmode](#) binary
- static const [fmtflags](#) boolalpha
- static const [seekdir](#) cur
- static const [fmtflags](#) dec
- static const [seekdir](#) end
- static const [iosstate](#) eofbit
- static const [iosstate](#) failbit
- static const [fmtflags](#) fixed
- static const [fmtflags](#) floatfield
- static const [iosstate](#) goodbit
- static const [fmtflags](#) hex
- static const [openmode](#) in
- static const [fmtflags](#) internal
- static const [fmtflags](#) left
- static const [fmtflags](#) oct
- static const [openmode](#) out
- static const [fmtflags](#) right
- static const [fmtflags](#) scientific
- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) & __loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename _ValueT >
[__istream_type](#) & [_M_extract](#) (_ValueT & __v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename _ValueT >
[__ostream_type](#) & [_M_insert](#) (_ValueT __v)
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) & __rhs) noexcept
- void [init](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
- void [move](#) ([basic_ios](#) & __rhs)
- void [move](#) ([basic_ios](#) & __rhs)
- void [set_rdbuf](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
- void [swap](#) ([basic_ios](#) & __rhs) noexcept
- void [swap](#) ([basic_istream](#) & __rhs)
- void [swap](#) ([basic_istream](#) & __rhs)
- void [swap](#) ([basic_ostream](#) & __rhs)

Protected Attributes

- `_Callback_list * _M_callbacks`
- `const __ctype_type * _M_ctype`
- `iosstate _M_exception`
- `char_type _M_fill`
- `bool _M_fill_init`
- `fmtflags _M_flags`
- `streamsize _M_gcount`
- `locale _M_ios_locale`
- `_Words _M_local_word [_S_local_word_size]`
- `const __num_get_type * _M_num_get`
- `const __num_put_type * _M_num_put`
- `streamsize _M_precision`
- `basic_streambuf<_CharT, _Traits> * _M_streambuf`
- `iosstate _M_streambuf_state`
- `basic_ostream<_CharT, _Traits> * _M_tie`
- `streamsize _M_width`
- `_Words * _M_word`
- `int _M_word_size`
- `_Words _M_word_zero`

6.263.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_stringstream<_CharT, _Traits, _Alloc>
```

Controlling input and output for `std::string`.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports reading from and writing to objects of type `std::basic_string`, using the inherited functions from `std::basic_istream`. To control the associated sequence, an instance of `std::basic_stringbuf` is used, which this page refers to as `sb`.

6.263.2 Member Typedef Documentation

`__num_put_type`

```
template<typename _CharT, typename _Traits>
typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]
```

These are non-standard types.

`event_callback`

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i) [inherited]
```

The type of an event callback function.

Parameters

<code>_e</code>	One of the members of the event enum.
<code>_b</code>	Reference to the <code>ios_base</code> object.
<code>_i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

iostate

```
typedef _Ios_Iostate std::ios_base::iostate [inherited]
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode [inherited]
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir [inherited]
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.263.3 Member Enumeration Documentation**event**

```
enum std::ios_base::event [inherited]
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.263.4 Constructor & Destructor Documentation

basic_stringstream() [1/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringstream< _CharT, _Traits, _Alloc >::basic_stringstream ( ) [inline]
```

Default constructor starts with an empty string buffer.

Initializes `sb` using the mode `in|out`, and passes `&sb` to the base class initializer. Does not allocate any buffer. That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_stringstream() [2/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringstream< _CharT, _Traits, _Alloc >::basic_stringstream (
    ios_base::openmode __m ) [inline], [explicit]
```

Starts with an empty string buffer.

Parameters

<code>__m</code>	Whether the buffer can read, or write, or both.
------------------	---

Initializes `sb` using the mode from `__m`, and passes `&sb` to the base class initializer. Does not allocate any buffer. That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

basic_stringstream() [3/3]

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringstream< _CharT, _Traits, _Alloc >::basic_stringstream (
    const __string_type & __str,
    ios_base::openmode __m = ios_base::out | ios_base::in ) [inline], [explicit]
```

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__m</code>	Whether the buffer can read, or write, or both.

Initializes `sb` using `__str` and `__m`, and passes `&sb` to the base class initializer. That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

~basic_stringstream()

```
template<typename _CharT , typename _Traits , typename _Alloc >
std::basic_stringstream< _CharT, _Traits, _Alloc >::~~basic_stringstream ( ) [inline]
```

The destructor does nothing.

The buffer is deallocated by the `stringbuf` object, not the formatting stream.

6.263.5 Member Function Documentation

_M_getloc()

```
const locale & std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by [std::money_get<_CharT, _Inlter >::do_get\(\)](#), [std::time_get<_CharT, _Inlter >::do_get\(\)](#), [std::num_get<_CharT, _Inlter >::do_get\(\)](#), [std::time_get<_CharT, _Inlter >::do_get_date\(\)](#), [std::time_get<_CharT, _Inlter >::do_get_monthname\(\)](#), [std::time_get<_CharT, _Inlter >::do_get_weekday\(\)](#), [std::time_get<_CharT, _Inlter >::do_get_year\(\)](#), [std::num_put<_CharT, _Outlter >::do_put\(\)](#), [std::time_put<_CharT, _Outlter >::do_put\(\)](#), [std::time_get<_CharT, _Inlter >::get\(\)](#), and [std::time_put<_CharT, _Outlter >::put\(\)](#).

bad()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::bad ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), and [std::basic_ios<_CharT, _Traits >::rdstate\(\)](#).

Referenced by [std::basic_ostream<_CharT, _Traits >::sentry::sentry\(\)](#).

clear()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::clear (
    iostate __state = goodbit ) [inherited]
```

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See [std::ios_base::iostate](#) for the possible bit values. Most users will not need to pass an argument.

Referenced by [std::basic_ios<_CharT, _Traits >::exceptions\(\)](#), [std::__detail::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits >::putb\(\)](#), [std::basic_istream<_CharT, _Traits >::seekg\(\)](#), [std::basic_istream<_CharT, _Traits >::seekg\(\)](#), [std::basic_ios<_CharT, _Traits >::setsbstate\(\)](#), and [std::basic_istream<_CharT, _Traits >::unget\(\)](#).

copyfmt()

```
template<typename _CharT , typename _Traits >
basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (
    const basic_ios< _CharT, _Traits > & __rhs ) [inherited]
```

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

References `std::__addressof()`, `std::basic_ios<_CharT, _Traits>::exceptions()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

eof()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios<_CharT, _Traits>::eof ( ) const [inline], [inherited]
Fast error checking.
```

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

References `std::ios_base::eofbit`, and `std::basic_ios<_CharT, _Traits>::rdstate()`.

exceptions() [1/2]

```
template<typename _CharT , typename _Traits >
iostate std::basic_ios<_CharT, _Traits>::exceptions ( ) const [inline], [inherited]
Throwing exceptions on errors.
```

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

exceptions() [2/2]

```
template<typename _CharT , typename _Traits >
void std::basic_ios<_CharT, _Traits>::exceptions (
    iostate __except ) [inline], [inherited]
Throwing exceptions on errors.
```

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
#include <iostream>
#include <fstream>
#include <exception>

int main()
{
```

```
std::set_terminate (__gnu_cxx::__verbose_terminate_handler);

std::ifstream f ("/etc/motd");

std::cerr << "Setting badbit\n";
f.setstate (std::ios_base::badbit);

std::cerr << "Setting exception mask\n";
f.exceptions (std::ios_base::badbit);
}
```

References [std::basic_ios<_CharT, _Traits>::clear\(\)](#).

fail()

```
template<typename _CharT , typename _Traits >
bool std::basic\_ios<\_CharT, \_Traits>::fail \( \) const [inline], [inherited]
```

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

References [std::ios_base::badbit](#), [std::ios_base::failbit](#), and [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::operator bool\(\)](#), [std::basic_ios<_CharT, _Traits>::operator!\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), and [std::basic_ostream<_CharT, _Traits>::tellg\(\)](#).

fill() [1/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \( \) const [inline], [inherited]
```

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::widen\(\)](#).

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), and [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

fill() [2/2]

```
template<typename _CharT , typename _Traits >
char_type std::basic\_ios<\_CharT, \_Traits>::fill \(
    char\_type \_\_ch ) [inline], [inherited]
```

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

References [std::basic_ios<_CharT, _Traits>::fill\(\)](#).

flags() [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline], [inherited]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator<<\(\), std::operator>>\(\), std::operator>>\(\), std::operator>>\(\), std::operator>>\(\), std::operator>>\(\), std::operator>>\(\), std::operator>>\(\), and std::operator>>\(\)](#).

flags() [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline], [inherited]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

flush()

```
template<typename _CharT , typename _Traits >
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::flush [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::flush\(\)](#).

gcount()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream<_CharT, _Traits>::gcount ( ) const [inline], [inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

get() [1/6]

```
template<typename _CharT , typename _Traits >
basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::get (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

get() [2/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb ) [inline], [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

get() [3/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    __streambuf_type & __sb,
    char_type __delim ) [inherited]
```

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#), [std::basic_streambuf< _CharT, _Traits >::snextc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::setstate\(\)](#).

get() [4/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream< _CharT, _Traits >::M_gcount](#), and [std::ios_base::goodbit](#).

get() [5/6]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

get() [6/6]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
------------------	----------------------

Parameters

<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

References [std::basic_istream<_CharT, _Traits >::_M_gcount](#), and [std::ios_base::goodbit](#).

getline() [1/3]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

getline() [2/3]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

getline() [3/3]

```
basic_istream< char > & std::basic_istream< char >::getline (
    char_type * __s,
    streamsize __n,
    char_type __delim ) [inherited]
```

Explicit specialization declarations, defined in `src/istream.cc`.

getloc()

```
locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), [std::operator>>\(\)](#), [std::operator>>\(\)](#), and [std::ws\(\)](#).

good()

```
template<typename _CharT , typename _Traits >
bool std::basic_ios< _CharT, _Traits >::good ( ) const [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

References [std::basic_ios<_CharT, _Traits>::rdstate\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::__detail::operator>>\(\)](#).

ignore() [1/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    void ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sbumpc\(\)](#).

ignore() [2/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n ) [inherited]
```

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), [std::ios_base::goodbit](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

ignore() [3/3]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (
    streamsize __n,
    int_type __delim ) [inherited]
```

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

References [std::basic_istream<_CharT, _Traits>::M_gcount](#), and [std::ios_base::goodbit](#).

imbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_ios< _CharT, _Traits >::imbue (
    const locale & __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

References `std::ios_base::imbue()`.

init()

```
template<typename _CharT , typename _Traits >
void std::basic_ios< _CharT, _Traits >::init (
    basic_streambuf< _CharT, _Traits > * __sb ) [protected], [inherited]
```

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Referenced by `std::basic_ios< _CharT, _Traits >::basic_ios()`.

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline], [inherited]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

narrow()

```
template<typename _CharT , typename _Traits >
char std::basic_ios< _CharT, _Traits >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

operator bool()

```
template<typename _CharT , typename _Traits >
```

```
std::basic_ios< _CharT, _Traits >::operator bool ( ) const [inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator"!()

```
template<typename _CharT , typename _Traits >
```

```
bool std::basic_ios< _CharT, _Traits >::operator! ( ) const [inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

References `std::basic_ios< _CharT, _Traits >::fail()`.

operator<<() [1/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [2/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    __ostream_type &(*) (__ostream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

operator<<() [3/17]

```
template<typename _CharT , typename _Traits >
```

```
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    __streambuf_type * __sb ) [inherited]
```

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

`operator<<()` [4/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

`operator<<()` [5/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    const void * __p ) [inline], [inherited]
```

Pointer arithmetic inserters.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [6/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [7/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    float __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [8/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    int __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

↔	A variable of builtin integral type.
<i>n</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [9/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits >::operator<< (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as std::endl and std::hex use these functions in constructs like "std::cout << std::endl". For more information, see the iomanip header.

operator<<() [10/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits >::operator<< (
    long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

\leftarrow	A variable of builtin integral type.
\leftarrow	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [11/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream<_CharT, _Traits >::operator<< (
    long double __f ) [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

\leftarrow	A variable of builtin floating point type.
\leftarrow	
\leftarrow	
\leftarrow	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [12/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [13/17]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::operator<< (
    short __n ) [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::num_put< _CharT, _Outiter >::put\(\)](#), and [std::basic_ios< _CharT, _Traits >](#)

operator<<() [14/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned int __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

operator<<() [15/17]

```
template<typename _CharT , typename _Traits >
```

```
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [16/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned long long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator<<() [17/17]

```
template<typename _CharT , typename _Traits >
__ostream_type & std::basic_ostream< _CharT, _Traits >::operator<< (
    unsigned short __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to perform numeric formatting.

operator>>() [1/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
```

```
__ios_type &(*) (__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `iosmanip` header.

operator>>() [2/17]

```
template<typename _CharT, typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    __istream_type &(*) (__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `iosmanip` header.

operator>>() [3/17]

```
template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    __streambuf_type * __sb ) [inherited]
```

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

operator>>() [4/17]

```
template<typename _CharT, typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    bool & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [5/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [6/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    float & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the num_get facet) to parse the input data.

operator>>() [7/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    int & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

$\ast this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [8/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    ios_base &(*) (ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

operator>>() [9/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

$_ \leftarrow$	A variable of builtin integral type.
$_n$	

Returns

$\ast this$ if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [10/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long double & __f ) [inline], [inherited]
```

Floating point arithmetic extractors.

Parameters

\leftarrow	A variable of builtin floating point type.
$_ \leftarrow$	
\leftarrow	
$_ \leftarrow$	
f	

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [11/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [12/17]

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (
    short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.


References [std::ios_base::badbit](#), [std::ios_base::failbit](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::setstate\(\)](#).

operator>>() [13/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned int & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [14/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [15/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned long long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful


These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [16/17]

```
template<typename _CharT , typename _Traits >
__istream_type & std::basic_istream< _CharT, _Traits >::operator>> (
    unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

 <code>__n</code>	A variable of builtin integral type.
--	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

operator>>() [17/17]

```
template<typename _CharT, typename _Traits>
__istream_type & std::basic_istream<_CharT, _Traits>::operator>> (
    void *& __p ) [inline], [inherited]
```

Basic arithmetic extractors.

Parameters

<code>__p</code>	A variable of pointer type.
------------------	-----------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

peek()

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::peek (
    void ) [inherited]
```

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline], [inherited]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`.

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

put()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (
    char_type __c ) [inherited]
```

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

*this

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), and [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#).

putback()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback (
    char_type __c ) [inherited]
```

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.


References [std::basic_istream< _CharT, _Traits >::M_gcount](#), [std::ios_base::badbit](#), [std::basic_ios< _CharT, _Traits >::clear\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), [std::basic_ios< _CharT, _Traits >::rdbuf\(\)](#), [std::basic_ios< _CharT, _Traits >::rdstate\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sputbackc\(\)](#).

pword()

```
void *& std::ios_base::pword (
    int __ix )    [inline], [inherited]
```

Access to void pointer array.

Parameters

 <code>__ix</code>	Index into the array.
---	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

rdbuf() [1/2]

```
template<typename _CharT , typename _Traits , typename _Alloc >
__stringbuf_type * std::basic_stringstream< _CharT, _Traits, _Alloc >::rdbuf ( ) const    [inline]
```

Accessing the underlying buffer.

Returns

The current basic_stringbuf buffer.

This hides both signatures of std::basic_ios::rdbuf().

rdbuf() [2/2]

```
template<typename _CharT , typename _Traits >
basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (
    basic_streambuf< _CharT, _Traits > * __sb )    [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
std::fstream    foo;           // or some other derived type
```

```
std::streambuf* p = .....;
foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

rdstate()

```
template<typename _CharT , typename _Traits >
iosstate std::basic_ios< _CharT, _Traits >::rdstate ( ) const [inline], [inherited]
```

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Referenced by `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::eof()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ios< _CharT, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::rdbuf()`.

read()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

readsome()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_istream< _CharT, _Traits >::readsome (
    char_type * __s,
    streamsize __n ) [inherited]
```

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rddbuf()->in_avail()`, called `A` here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_ios<_CharT, _Traits>::rddbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index ) [inherited]
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

seekg() [1/2]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (
    off_type __off,
    ios_base::seekdir __dir ) [inherited]
```

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekg() [2/2]

```
template<typename _CharT, typename _Traits>
basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (
    pos_type __pos) [inherited]
```

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::basic_ios<_CharT, _Traits>::clear\(\)](#), [std::ios_base::eofbit](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::ios_base::in](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [1/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (
    off_type __off,
    ios_base::seekdir __dir) [inherited]
```

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

seekp() [2/2]

```
template<typename _CharT, typename _Traits>
basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (
    pos_type __pos) [inherited]
```

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

References [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::failbit](#), [std::ios_base::out](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Referenced by [std::__detail::operator>>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask) [inline], [inherited]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

setstate()

```
template<typename _CharT, typename _Traits >
void std::basic_ios< _CharT, _Traits >::setstate (
    iostate __state) [inline], [inherited]
```

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

References `std::basic_ios< _CharT, _Traits >::clear()`, and `std::basic_ios< _CharT, _Traits >::rdstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::getline()`, `std::getline()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

str() [1/2]

```
template<typename _CharT, typename _Traits, typename _Alloc >
__string_type std::basic_stringstream< _CharT, _Traits, _Alloc >::str ( ) const [inline]
```

Copying out the string buffer.

Returns

`rdbuf() -> str()`

str() [2/2]

```
template<typename _CharT, typename _Traits, typename _Alloc >
void std::basic_stringstream< _CharT, _Traits, _Alloc >::str (
    const __string_type & __s) [inline]
```

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf() -> str(s)`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_istream< _CharT, _Traits >::sync (
    void ) [inherited]
```

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

References [std::ios_base::badbit](#), [std::ios_base::goodbit](#), [std::basic_streambuf<_CharT, _Traits>::pubsync\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), and [std::basic_ios<_CharT, _Traits>::setstate\(\)](#).

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static], [inherited]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

tellg()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits >::pos_type std::basic_istream< _CharT, _Traits >::tellg (
    void ) [inherited]
```

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

References [std::ios_base::badbit](#), [std::ios_base::cur](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::ios_base::in](#), and [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#).

tellp()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits >::pos_type std::basic_ostream< _CharT, _Traits >::tellp [inherited]
```

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

References `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::out`, and `std::basic_ios< _CharT, _Traits >::rdbuf()`.

tie() [1/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie ( ) const [inline],
[inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ios< _CharT, _Traits >::copyfmt()`.

tie() [2/2]

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::tie (
    basic_ostream< _CharT, _Traits > * __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

unget()

```
template<typename _CharT , typename _Traits >
basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::unget (
    void ) [inherited]
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf() -> sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_streambuf< _CharT, _Traits >::sungetc()`.

Referenced by `std::__detail::operator>>()`.

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

widen()

```
template<typename _CharT , typename _Traits >
char_type std::basic_ios< _CharT, _Traits >::widen (
    char __c ) const [inline], [inherited]
```

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Referenced by `std::basic_ios< _CharT, _Traits >::fill()`, `std::getline()`, `std::getline()`, `std::tr2::operator>>()`, and `std::basic_ostream< char, _Traits >::put()`.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline], [inherited]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline], [inherited]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of width().

write()

```
template<typename _CharT , typename _Traits >
basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (
    const char_type * __s,
    streamsize __n ) [inherited]
```

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

*this

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, badbit will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static], [inherited]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.263.6 Member Data Documentation

M_gcount

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected], [inherited]
```

The number of characters extracted in the previous unformatted function; see `gcount()`.

Referenced by `std::basic_istream< char >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< char >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< char >::seekg()`, `std::basic_istream< char >::sync()`, `std::basic_istream< char >::unget()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

adjustfield

```
const fmtflags std::ios_base::adjustfield [static], [inherited]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put< _CharT, _Outiter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static], [inherited]
```

Seek to end before each write.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

ate

```
const openmode std::ios_base::ate [static], [inherited]
```

Open and seek to end immediately after opening.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

badbit

```
const iostate std::ios_base::badbit [static], [inherited]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_ios< _CharT, _Traits >::bad()`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< char >::get()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_ostream< char, _Traits >::operator<<()`, `std::operator>>()`, `std::basic_istream< char >::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< char >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

basefield

```
const fmtflags std::ios_base::basefield [static], [inherited]
```

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by `std::dec()`, `std::num_get< _CharT, _Initer >::do_get()`, `std::num_put< _CharT, _Outiter >::do_put()`, `std::hex()`, and `std::oct()`.

beg

```
const seekdir std::ios_base::beg [static], [inherited]
```


failbit

```
const ios_base::failbit [static], [inherited]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_ostream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ios<_CharT, _Traits>::fail\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<char>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::read\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), and [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#).

fixed

```
const fmtflags std::ios_base::fixed [static], [inherited]
```

Generate floating-point output in fixed-point notation.

Referenced by [std::fixed\(\)](#), and [std::hexfloat\(\)](#).

floatfield

```
const fmtflags std::ios_base::floatfield [static], [inherited]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const ios_base::goodbit [static], [inherited]
```

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_ostream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unget\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static], [inherited]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _InIter>::do_get\(\)](#), [std::num_put<_CharT, _OutIter>::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static], [inherited]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#), and [std::ws\(\)](#).

`std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_stringbuf< _CharT, _Traits, _`
and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

internal

```
const fmtflags std::ios_base::internal [static], [inherited]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by `std::internal()`.

left

```
const fmtflags std::ios_base::left [static], [inherited]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, and `std::left()`.

oct

```
const fmtflags std::ios_base::oct [static], [inherited]
```

Converts integer input or generates integer output in octal base.

Referenced by `std::oct()`.

out

```
const openmode std::ios_base::out [static], [inherited]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

right

```
const fmtflags std::ios_base::right [static], [inherited]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by `std::right()`.

scientific

```
const fmtflags std::ios_base::scientific [static], [inherited]
```

Generates floating-point output in scientific notation.

Referenced by `std::hexfloat()`, and `std::scientific()`.

showbase

```
const fmtflags std::ios_base::showbase [static], [inherited]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

showpoint

```
const fmtflags std::ios_base::showpoint [static], [inherited]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

showpos

```
const fmtflags std::ios_base::showpos [static], [inherited]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static], [inherited]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static], [inherited]
```

Truncate an existing stream when opening. Default for ofstream.

unitbuf

```
const fmtflags std::ios_base::unitbuf [static], [inherited]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmtflags std::ios_base::uppercase [static], [inherited]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

6.264 std::bernoulli_distribution Class Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef bool [result_type](#)

Public Member Functions

- [bernoulli_distribution](#) ()
- [bernoulli_distribution](#) (const [param_type](#) &__p)
- [bernoulli_distribution](#) (double __p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)

- `template<typename _UniformRandomNumberGenerator >`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `double p () const`
- `param_type param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `bool operator== (const bernoulli_distribution &__d1, const bernoulli_distribution &__d2)`

6.264.1 Detailed Description

A Bernoulli random number distribution.

Generates a sequence of true and false values with likelihood p that true will come up and $(1 - p)$ that false will appear.

6.264.2 Member Typedef Documentation

result_type

```
typedef bool std::bernoulli_distribution::result_type
```

The type of the range of the distribution.

6.264.3 Constructor & Destructor Documentation

bernoulli_distribution() [1/2]

```
std::bernoulli_distribution::bernoulli_distribution ( ) [inline]
```

Constructs a Bernoulli distribution with likelihood 0.5.

bernoulli_distribution() [2/2]

```
std::bernoulli_distribution::bernoulli_distribution (
    double __p ) [inline], [explicit]
```

Constructs a Bernoulli distribution with likelihood p .

Parameters

<code>__p</code>	[IN] The likelihood of a true result being returned. Must be in the interval $[0, 1]$.
------------------	---

6.264.4 Member Function Documentation

max()

```
result_type std::bernoulli_distribution::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.

References [std::numeric_limits<_Tp>::max\(\)](#).

min()

```
result_type std::bernoulli_distribution::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

References [std::numeric_limits<_Tp>::min\(\)](#).

operator>()

```
template<typename _UniformRandomNumberGenerator >
result_type std::bernoulli_distribution::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [operator\(\)\(\)](#).

Referenced by [operator\(\)\(\)](#).

p()

```
double std::bernoulli_distribution::p ( ) const [inline]
```

Returns the `p` parameter of the distribution.

param() [1/2]

```
param_type std::bernoulli_distribution::param ( ) const [inline]
```

Returns the parameter set of the distribution.

Referenced by [std::operator>>\(\)](#).

param() [2/2]

```
void std::bernoulli_distribution::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
void std::bernoulli_distribution::reset ( ) [inline]
```

Resets the distribution state.

Does nothing for a Bernoulli distribution.

6.264.5 Friends And Related Symbol Documentation

operator==

```
bool operator== (
    const bernoulli_distribution & __d1,
    const bernoulli_distribution & __d2 ) [friend]
```

Return true if two Bernoulli distributions have the same parameters.

The documentation for this class was generated from the following files:

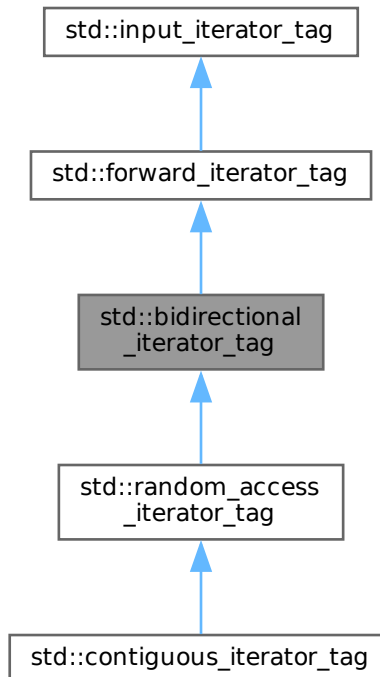
- [random.h](#)

- [bits/random.tcc](#)

6.265 std::bidirectional_iterator_tag Struct Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for std::bidirectional_iterator_tag:



6.265.1 Detailed Description

Bidirectional iterators support a superset of forward iterator operations.

The documentation for this struct was generated from the following file:

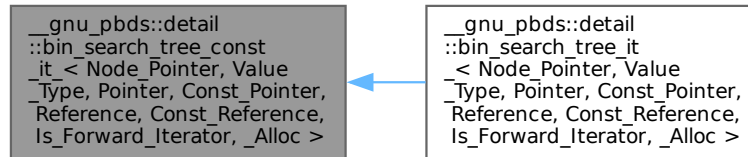
- [stl_iterator_base_types.h](#)

6.266 __gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > Class Template Reference

```
#include <point_iterators.hpp>
```

Inheritance diagram for __gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_↵

Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >:



Public Types

- typedef Const_Pointer **const_pointer**
- typedef Const_Reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef Pointer **pointer**
- typedef Reference **reference**
- typedef Value_Type **value_type**

Public Member Functions

- **bin_search_tree_const_it_** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other)
- **bin_search_tree_const_it_** (const Node_Pointer p_nd=0)
- bool **operator!=** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other) const
- bool **operator!=** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const
- const_reference **operator*** () const
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator++** ()
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **operator++** (int)
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator--** ()
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **operator--** (int)
- const_pointer **operator->** () const
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other)
- [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other)
- bool **operator==** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other) const
- bool **operator==** (const [bin_search_tree_const_it_](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const

Public Attributes

- Node_Parser **m_p_nd**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

6.266.1 Detailed Description

```
template<typename Node_Parser, typename Value_Type, typename Parser, typename Const_Parser, type-
name Reference, typename Const_Reference, bool Is_Forward_Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_const_it_< Node_Parser, Value_Type, Parser, Const_Parser, Ref-
erence, Const_Reference, Is_Forward_Iterator, _Alloc >
```

Const iterator.

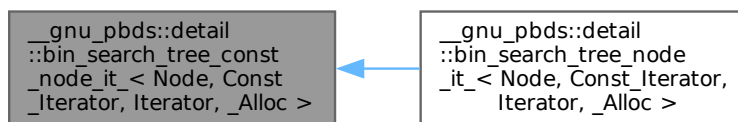
The documentation for this class was generated from the following file:

- [point_iterators.hpp](#)

6.267 __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > Class Template Reference

```
#include <node_iterators.hpp>
```

Inheritance diagram for __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >:



Public Types

- typedef Const_Iterator [const_reference](#)
- typedef [trivial_iterator_difference_type](#) [difference_type](#)
- typedef [trivial_iterator_tag](#) [iterator_category](#)
- typedef [rebind_traits](#)< _Alloc, [metadata_type](#) >::const_reference [metadata_const_reference](#)
- typedef Node::metadata_type [metadata_type](#)
- typedef Const_Iterator [reference](#)
- typedef Const_Iterator [value_type](#)

Public Member Functions

- `bin_search_tree_const_node_it_` (const node_pointer p_nd=0)
- `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` `get_l_child` () const
- `metadata_const_reference` `get_metadata` () const
- `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` `get_r_child` () const
- bool `operator!=` (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const
- `const_reference` `operator*` () const
- bool `operator==` (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const

Public Attributes

- node_pointer `m_p_nd`

6.267.1 Detailed Description

```
template<typename Node, class Const_Iterator, class Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >
```

Const node iterator.

6.267.2 Member Typedef Documentation

`const_reference`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Const_Iterator __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::const_reference
Iterator's __const reference type.
```

`difference_type`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef trivial_iterator_difference_type __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::difference_type
Difference type.
```

`iterator_category`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef trivial_iterator_tag __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, Iterator, _Alloc >::iterator_category
Category.
```

`metadata_const_reference`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef rebind_traits<_Alloc,metadata_type>::const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::metadata_const_reference
Const metadata reference type.
```

`metadata_type`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Node::metadata_type __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, Iterator, _Alloc >::metadata_type
Metadata type.
```


reference

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Const_Iterator __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Const_Iterator __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::value_type
Iterator's value type.
```

6.267.3 Member Function Documentation

get_l_child()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::get_l_child ( ) const [inline]
Returns the __const node iterator associated with the left node.
```

get_metadata()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
metadata_const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::get_metadata ( ) const [inline]
Metadata access.
```

get_r_child()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::get_r_child ( ) const [inline]
Returns the __const node iterator associated with the right node.
```

operator"!="()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bool __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::operator!= (
    const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > &
    other ) const [inline]
Compares (negatively) to a different iterator object.
```

operator*()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator,
_Alloc >::operator* ( ) const [inline]
Access.
```

operator==()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bool __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc
>::operator==(
    const bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc > &
    other ) const [inline]
```

Compares to a different iterator object.

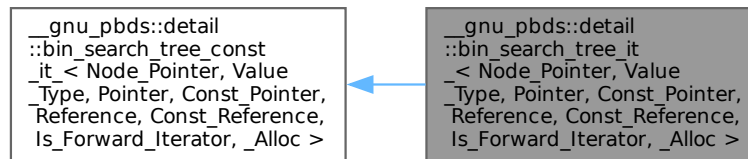
The documentation for this class was generated from the following file:

- [bin_search_tree_/node_iterators.hpp](#)

6.268 `__gnu_pbds::detail::bin_search_tree_it< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` Class Template Reference

```
#include <point_iterators.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_it< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >`:



Public Types

- typedef Const_Pointer **const_pointer**
- typedef Const_Reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef Pointer **pointer**
- typedef Reference **reference**
- typedef Value_Type **value_type**

Public Member Functions

- **bin_search_tree_it** (const [bin_search_tree_it](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other)
- **bin_search_tree_it** (const Node_Pointer p_nd=0)
- bool **operator!=** (const [bin_search_tree_const_it](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, !Is_Forward_Iterator, _Alloc > &other) const
- bool **operator!=** (const [bin_search_tree_const_it](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const
- [bin_search_tree_const_it](#)< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >::reference **operator*** () const

- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator++** ()
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **operator++** (int)
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator--** ()
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **operator--** (int)
- [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >::pointer **operator->** () const
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other)
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other)
- bool **operator==** (const [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const
- bool **operator==** (const [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const

Public Attributes

- Node_Pointer **m_p_nd**

Protected Types

- typedef [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **base_it_type**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

6.268.1 Detailed Description

template<typename Node_Pointer, typename Value_Type, typename Pointer, typename Const_Pointer, typename Reference, typename Const_Reference, bool Is_Forward_Iterator, typename _Alloc>
class [__gnu_pbds::detail::bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >

Iterator.

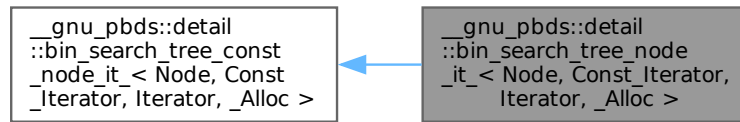
The documentation for this class was generated from the following file:

- [point_iterators.hpp](#)

6.269 [__gnu_pbds::detail::bin_search_tree_node_it](#) < Node, Const_Iterator, Iterator, _Alloc > Class Template Reference

```
#include <node_iterators.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >`:



Public Types

- typedef Iterator [const_reference](#)
- typedef [trivial_iterator_difference_type](#) difference_type
- typedef [trivial_iterator_tag](#) iterator_category
- typedef [rebind_traits< _Alloc, metadata_type >::const_reference](#) metadata_const_reference
- typedef Node::metadata_type metadata_type
- typedef Iterator [reference](#)
- typedef Iterator [value_type](#)

Public Member Functions

- `bin_search_tree_node_it_` (const node_pointer p_nd=0)
- `bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >` [get_l_child](#) () const
- `metadata_const_reference` [get_metadata](#) () const
- `bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >` [get_r_child](#) () const
- bool [operator!=](#) (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const
- Iterator [operator*](#) () const
- bool [operator==](#) (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const

Public Attributes

- node_pointer `m_p_nd`

6.269.1 Detailed Description

```

template<typename Node, class Const_Iterator, class Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >

```

Node iterator.

6.269.2 Member Typedef Documentation

`const_reference`

```

template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Iterator __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _↵
_Alloc >::const_reference

```

Iterator's `__const` reference type.

difference_type

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef trivial_iterator_difference_type __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::difference_type [inherited]
Difference type.
```

iterator_category

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef trivial_iterator_tag __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_↵
Iterator, Iterator, _Alloc >::iterator_category [inherited]
Category.
```

metadata_const_reference

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef rebind_traits<_Alloc,metadata_type>::const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::metadata_const_reference [inherited]
Const metadata reference type.
```

metadata_type

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Node::metadata_type __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_↵
Iterator, Iterator, _Alloc >::metadata_type [inherited]
Metadata type.
```

reference

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Iterator __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _↵
Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
typedef Iterator __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _↵
Alloc >::value_type
Iterator's value type.
```

6.269.3 Member Function Documentation

get_l_child()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc > __gnu_pbds::detail::bin_search_tree_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::get_l_child ( ) const [inline]
Returns the node iterator associated with the left node.
```

get_metadata()

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
metadata_const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::get_metadata ( ) const [inline], [inherited]
```

Metadata access.

`get_r_child()`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc > __gnu_pbds::detail::bin_search_tree_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::get_r_child ( ) const [inline]
```

Returns the node iterator associated with the right node.

`operator!=(())`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bool __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::operator!= (
    const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > &
    other ) const [inline], [inherited]
```

Compares (negatively) to a different iterator object.

`operator*()`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
Iterator __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >↵
::operator* ( ) const [inline]
```

Access.

`operator==(())`

```
template<typename Node , class Const_Iterator , class Iterator , typename _Alloc >
bool __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::operator== (
    const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > &
    other ) const [inline], [inherited]
```

Compares to a different iterator object.

The documentation for this class was generated from the following file:

- [bin_search_tree_/node_iterators.hpp](#)

6.270 `__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >` Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **const_reverse_iterator**
- typedef `Node` **node**
- typedef `bin_search_tree_const_node_it_< Node, point_const_iterator, point_iterator, _Alloc >` **node_const_iterator**
- typedef `bin_search_tree_node_it_< Node, point_const_iterator, point_iterator, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node`↵
_update_pointer

- typedef [bin_search_tree_const_it](#) < typename node_alloc_traits::pointer, typename [type_traits::value_type](#), typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > **point_const_iterator**
- typedef [bin_search_tree_it](#) < typename node_alloc_traits::pointer, typename [type_traits::value_type](#), typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > **point_iterator**
- typedef [bin_search_tree_it](#) < typename node_alloc_traits::pointer, typename [type_traits::value_type](#), typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc > **reverse_iterator**

6.270.1 Detailed Description

```
template<typename Key, typename Mapped, class Cmp_Fn, template< typename Node_CIttr, class Node_Itr,
class _Cmp_Fn, typename _Alloc > class Node_Update, class Node, typename _Alloc>
struct __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >
```

Binary search tree traits, primary template.

6.270.2 Member Typedef Documentation

node_const_iterator

```
template<typename Key , typename Mapped , class Cmp_Fn , template< typename Node_CIttr, class
Node_Itr, class _Cmp_Fn, typename _Alloc > class Node_Update, class Node , typename _Alloc >
typedef bin\_search\_tree\_const\_node\_it< Node, point\_const\_iterator, point\_iterator, _Alloc> \_\_gnu\_pbds::detail::bin\_search\_tree\_traits
<Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

The documentation for this struct was generated from the following file:

- [bin_search_tree_/traits.hpp](#)

6.271 [__gnu_pbds::detail::bin_search_tree_traits](#)< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef [bin_search_tree_const_it](#) < typename node_alloc_traits::pointer, typename [type_traits::value_type](#), typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc > **const_reverse_iterator**
- typedef Node **node**
- typedef [bin_search_tree_const_node_it](#) < Node, [point_const_iterator](#), [point_iterator](#), _Alloc > [node_const_iterator](#)
- typedef [node_const_iterator](#) **node_iterator**
- typedef Node_Update< [node_const_iterator](#), [node_iterator](#), Cmp_Fn, _Alloc > **node_update**
- typedef [__gnu_pbds::null_node_update](#)< [node_const_iterator](#), [node_iterator](#), Cmp_Fn, _Alloc > * **null_node_update_pointer**
- typedef [bin_search_tree_const_it](#) < typename node_alloc_traits::pointer, typename [type_traits::value_type](#), typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > **point_const_iterator**
- typedef [point_const_iterator](#) **point_iterator**
- typedef [const_reverse_iterator](#) **reverse_iterator**

6.271.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_CIttr, class Node_Itr, class _Cmp_Fn, type-
name _Alloc > class Node_Update, class Node, typename _Alloc>
struct __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >
```

Specialization.

6.271.2 Member Typedef Documentation

`node_const_iterator`

```
template<typename Key , class Cmp_Fn , template< typename Node_CIttr, class Node_Itr, class _Cmp_Fn, type-
name _Alloc > class Node_Update, class Node , typename _Alloc >
typedef bin_search_tree_const_node_it< Node, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits<
Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >::node_const_iterator
```

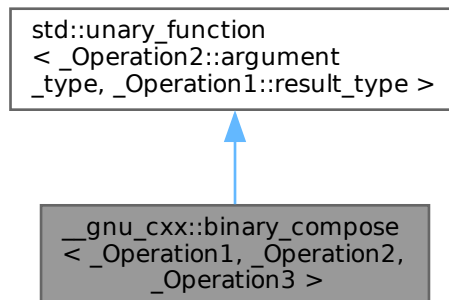
This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

The documentation for this struct was generated from the following file:

- [bin_search_tree_/traits.hpp](#)

6.272 `__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >` Class Template Reference

Inheritance diagram for `__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >`:



Public Types

- typedef `_Operation2::argument_type` [argument_type](#)
- typedef `_Operation1::result_type` [result_type](#)

Public Member Functions

- **`binary_compose`** (`const _Operation1 &__x, const _Operation2 &__y, const _Operation3 &__z`)
- `_Operation1::result_type` **`operator()`** (`const typename _Operation2::argument_type &__x`) const

Protected Attributes

- `_Operation1 _M_fn1`
- `_Operation2 _M_fn2`
- `_Operation3 _M_fn3`

6.272.1 Detailed Description

```
template<class _Operation1, class _Operation2, class _Operation3>
class __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >
```

An [SGI extension](#) .

6.272.2 Member Typedef Documentation

`argument_type`

```
typedef _Operation2::argument_type std::unary\_function< _Operation2::argument_type , _Operation1↵
::result_type >::argument_type [inherited]
argument_type is the type of the argument
```

`result_type`

```
typedef _Operation1::result_type std::unary\_function< _Operation2::argument_type , _Operation1↵
::result_type >::result_type [inherited]
result_type is the return type
```

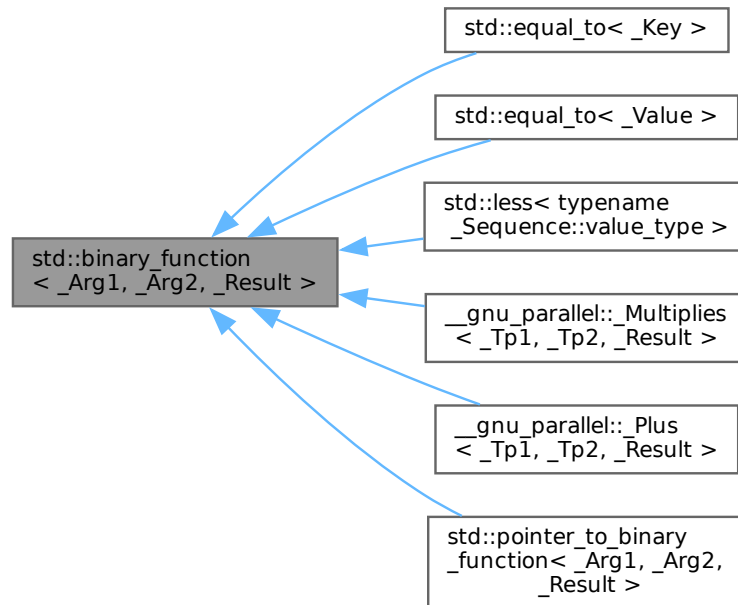
The documentation for this class was generated from the following file:

- [ext/functional](#)

6.273 `std::binary_function< _Arg1, _Arg2, _Result >` Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::binary_function< _Arg1, _Arg2, _Result >:



Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

6.273.1 Detailed Description

template<typename `_Arg1`, typename `_Arg2`, typename `_Result`>

struct `std::binary_function< _Arg1, _Arg2, _Result >`

Helper for defining adaptable binary function objects.

Deprecated Deprecated in C++11, no longer in the standard since C++17.

6.273.2 Member Typedef Documentation

`first_argument_type`

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg1 std::binary_function< _Arg1, _Arg2, _Result >::first_argument_type
first_argument_type is the type of the first argument
```

`result_type`

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Result std::binary_function< _Arg1, _Arg2, _Result >::result_type
result_type is the return type
```

second_argument_type

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg2 std::binary\_function< _Arg1, _Arg2, _Result >::second_argument_type
```

`second_argument_type` is the type of the second argument

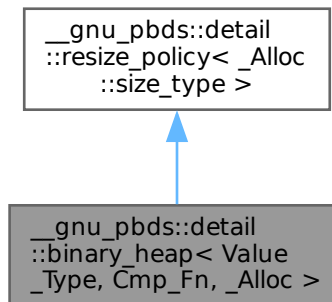
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.274 __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference

```
#include <binary_heap_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >`:

**Public Types**

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `cond_dealtor< value_type, _Alloc >` **cond_dealtor_t**
- typedef `binary_heap_const_iterator< value_type, entry, simple_value, _Alloc >` **const_iterator**
- typedef `__rebind_v::const_pointer` **const_pointer**
- typedef `__rebind_v::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `__conditional_type< simple_value, value_type, pointer >::__type` **entry**
- typedef `rebind_traits< _Alloc, entry >::allocator_type` **entry_allocator**
- typedef `entry_cmp< Value_Type, Cmp_Fn, _Alloc, is_simple< Value_Type >::value >::type` **entry_cmp**
- typedef `rebind_traits< _Alloc, entry >::pointer` **entry_pointer**
- typedef `const_iterator` **iterator**
- typedef `binary_heap_point_const_iterator< value_type, entry, simple_value, _Alloc >` **point_const_iterator**
- typedef `point_const_iterator` **point_iterator**
- typedef `__rebind_v::pointer` **pointer**
- typedef `__rebind_v::reference` **reference**
- typedef `__gnu_pbds::detail::resize_policy< typename _Alloc::size_type >` **resize_policy**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **binary_heap** (const [binary_heap](#) &)
- **binary_heap** (const cmp_fn &)
- [iterator](#) **begin** ()
- [const_iterator](#) **begin** () const
- void **clear** ()
- bool **empty** () const
- [iterator](#) **end** ()
- [const_iterator](#) **end** () const
- void **erase** ([point_iterator](#))
- void **erase_at** (entry_pointer, size_type, false_type)
- void **erase_at** (entry_pointer, size_type, true_type)
- template<typename Pred >
size_type **erase_if** (Pred)
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- size_type **get_new_size_for_arbitrary** (size_type) const
- size_type **get_new_size_for_grow** () const
- size_type **get_new_size_for_shrink** () const
- bool **grow_needed** (size_type) const
- void **join** ([binary_heap](#) &)
- size_type **max_size** () const
- void **modify** ([point_iterator](#), const_reference)
- void **notify_arbitrary** (size_type)
- void **notify_grow_resize** ()
- void **notify_shrink_resize** ()
- void **pop** ()
- [point_iterator](#) **push** (const_reference)
- bool **resize_needed_for_grow** (size_type) const
- bool **resize_needed_for_shrink** (size_type) const
- bool **shrink_needed** (size_type) const
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, [binary_heap](#) &)
- void **swap** ([binary_heap](#) &)
- void **swap** ([resize_policy](#)< _Alloc::size_type > &)
- const_reference **top** () const

Static Public Attributes

- static const _Alloc::size_type **min_size**

Protected Member Functions

- template<typename It >
void **copy_from_range** (It, It)

6.274.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >
```

Binary heaps composed of resize and compare policies.

Based on CLRS.

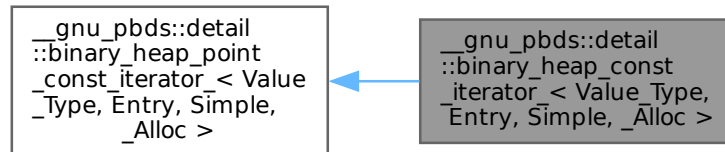
The documentation for this class was generated from the following file:

- [binary_heap.hpp](#)

6.275 __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > Class Template Reference

```
#include <const_iterator.hpp>
```

Inheritance diagram for __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >:



Public Types

- typedef [base_type::const_pointer](#) const_pointer
- typedef [base_type::const_reference](#) const_reference
- typedef [_Alloc::difference_type](#) difference_type
- typedef [std::forward_iterator_tag](#) iterator_category
- typedef [base_type::pointer](#) pointer
- typedef [base_type::reference](#) reference
- typedef [base_type::value_type](#) value_type

Public Member Functions

- [binary_heap_const_iterator_\(\)](#)
- [binary_heap_const_iterator_\(const binary_heap_const_iterator_ &other\)](#)
- [binary_heap_const_iterator_\(entry_pointer p_e\)](#)
- bool [operator!=](#)(const [binary_heap_const_iterator_](#) &other) const
- bool [operator!=](#)(const [binary_heap_point_const_iterator_](#) &other) const
- [const_reference operator*](#) () const
- [binary_heap_const_iterator_ & operator++](#) ()
- [binary_heap_const_iterator_ operator++](#) (int)
- [const_pointer operator->](#) () const
- bool [operator==](#)(const [binary_heap_const_iterator_](#) &other) const
- bool [operator==](#)(const [binary_heap_point_const_iterator_](#) &other) const

Public Attributes

- entry_pointer `m_p_e`

6.275.1 Detailed Description

```
template<typename Value_Type, typename Entry, bool Simple, typename _Alloc>
class __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >
```

Const point-type iterator.

6.275.2 Member Typedef Documentation

`const_pointer`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef base_type::const_pointer __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type,
Entry, Simple, _Alloc >::const_pointer
Iterator's const pointer type.
```

`const_reference`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef base_type::const_reference __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type,
Entry, Simple, _Alloc >::const_reference
Iterator's const reference type.
```

`difference_type`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef _Alloc::difference_type __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type,
Entry, Simple, _Alloc >::difference_type
Difference type.
```

`iterator_category`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef std::forward_iterator_tag __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type,
Entry, Simple, _Alloc >::iterator_category
Category.
```

`pointer`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef base_type::pointer __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry,
Simple, _Alloc >::pointer
Iterator's pointer type.
```

`reference`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef base_type::reference __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry,
Simple, _Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef base\_type::value\_type \_\_gnu\_pbds::detail::binary\_heap\_const\_iterator\_< Value_Type, Entry,
Simple, _Alloc >::value_type
Iterator's value type.
```

6.275.3 Constructor & Destructor Documentation**binary_heap_const_iterator_()** [1/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
\_\_gnu\_pbds::detail::binary\_heap\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >::binary_↵
heap_const_iterator_ ( ) [inline]
Default constructor.
```

binary_heap_const_iterator_() [2/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
\_\_gnu\_pbds::detail::binary\_heap\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >::binary_↵
heap_const_iterator_ (
    const binary\_heap\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc > & other )
[inline]
Copy constructor.
```

6.275.4 Member Function Documentation**operator"!="()** [1/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool \_\_gnu\_pbds::detail::binary\_heap\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >::operator!=
(
    const binary\_heap\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc > & other )
const [inline]
Compares content (negatively) to a different iterator object.
```

operator"!="() [2/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >↵
::operator!= (
    const binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc > & other
) const [inline], [inherited]
Compares content (negatively) to a different iterator object.
```

operator*()

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
const\_reference \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple,
_Alloc >::operator* ( ) const [inline], [inherited]
Access.
```

operator->()

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
const\_pointer \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, ↵
_Alloc >::operator-> ( ) const [inline], [inherited]
```

Access.

`operator==()` [1/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator==
(
    const binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other )
const [inline]
```

Compares content to a different iterator object.

`operator==()` [2/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >↵
::operator== (
    const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other
) const [inline], [inherited]
```

Compares content to a different iterator object.

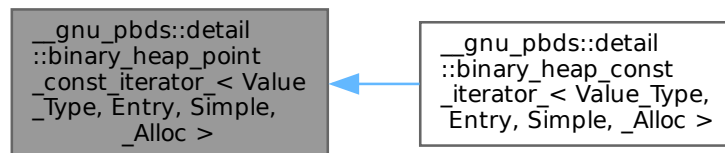
The documentation for this class was generated from the following file:

- [binary_heap_/const_iterator.hpp](#)

6.276 `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >` Class Template Reference

```
#include <point_const_iterator.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >`:



Public Types

- typedef [rebind_traits](#)< [_Alloc](#), [value_type](#) >::const_pointer [const_pointer](#)
- typedef [rebind_traits](#)< [_Alloc](#), [value_type](#) >::const_reference [const_reference](#)
- typedef [trivial_iterator_difference_type](#) [difference_type](#)
- typedef [trivial_iterator_tag](#) [iterator_category](#)
- typedef [rebind_traits](#)< [_Alloc](#), [value_type](#) >::pointer [pointer](#)
- typedef [rebind_traits](#)< [_Alloc](#), [value_type](#) >::reference [reference](#)
- typedef [Value_Type](#) [value_type](#)

Public Member Functions

- [binary_heap_point_const_iterator_\(\)](#)
- [binary_heap_point_const_iterator_\(const binary_heap_point_const_iterator_ &other\)](#)
- [binary_heap_point_const_iterator_\(entry_pointer p_e\)](#)
- [bool operator!= \(const binary_heap_point_const_iterator_ &other\) const](#)
- [const_reference operator* \(\) const](#)
- [const_pointer operator-> \(\) const](#)
- [bool operator== \(const binary_heap_point_const_iterator_ &other\) const](#)

Public Attributes

- entry_pointer [m_p_e](#)

Protected Types

- typedef [rebind_traits<_Alloc, Entry >::pointer](#) [entry_pointer](#)

6.276.1 Detailed Description

```
template<typename Value_Type, typename Entry, bool Simple, typename _Alloc>
class __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >
```

Const point-type iterator.

6.276.2 Member Typedef Documentation

const_pointer

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef rebind\_traits<\_Alloc,value\_type>::const\_pointer \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_<
Value_Type, Entry, Simple, _Alloc >::const_pointer
```

Iterator's const pointer type.

const_reference

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef rebind\_traits<\_Alloc,value\_type>::const\_reference \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_<
Value_Type, Entry, Simple, _Alloc >::const_reference
```

Iterator's const reference type.

difference_type

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef trivial\_iterator\_difference\_type \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_<
Value_Type, Entry, Simple, _Alloc >::difference_type
```

Difference type.

iterator_category

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef trivial\_iterator\_tag \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type,
Entry, Simple, _Alloc >::iterator_category
```

Category.

pointer

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef rebind\_traits<_Alloc,value_type>::pointer \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_<
Value_Type, Entry, Simple, _Alloc >::pointer
Iterator's pointer type.
```

reference

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef rebind\_traits<_Alloc,value_type>::reference \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_<
Value_Type, Entry, Simple, _Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
typedef Value_Type \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry,
Simple, _Alloc >::value_type
Iterator's value type.
```

6.276.3 Constructor & Destructor Documentation

`binary_heap_point_const_iterator_()` [1/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
\_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >↵
::binary\_heap\_point\_const\_iterator\_ ( ) [inline]
Default constructor.
```

`binary_heap_point_const_iterator_()` [2/2]

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
\_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >↵
::binary\_heap\_point\_const\_iterator\_ (
    const binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc > & other
) [inline]
Copy constructor.
```

6.276.4 Member Function Documentation

`operator"!="()`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc >↵
::operator!= (
    const binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple, _Alloc > & other
) const [inline]
Compares content (negatively) to a different iterator object.
```

`operator*()`

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
const_reference \_\_gnu\_pbds::detail::binary\_heap\_point\_const\_iterator\_< Value_Type, Entry, Simple,
_Alloc >::operator* ( ) const [inline]
Access.
```

operator->()

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
const_pointer __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, ↔
_Alloc >::operator-> ( ) const [inline]
Access.
```

operator==()

```
template<typename Value_Type , typename Entry , bool Simple, typename _Alloc >
bool __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >↔
::operator== (
    const binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc > & other
) const [inline]
```

Compares content to a different iterator object.

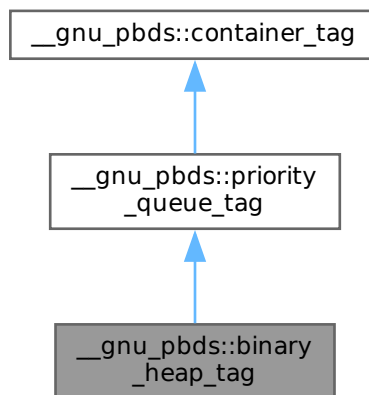
The documentation for this class was generated from the following file:

- [binary_heap_/point_const_iterator.hpp](#)

6.277 __gnu_pbds::binary_heap_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::binary_heap_tag:

**6.277.1 Detailed Description**

Binary-heap (array-based).

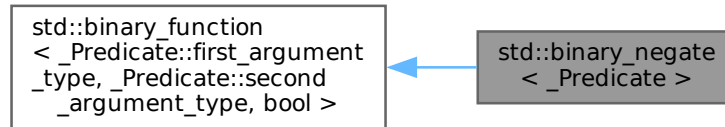
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.278 std::binary_negate<_Predicate > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::binary_negate<_Predicate>:



Public Types

- typedef _Predicate::first_argument_type [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Predicate::second_argument_type [second_argument_type](#)

Public Member Functions

- constexpr **binary_negate** (const _Predicate &__x)
- constexpr bool **operator()** (const typename _Predicate::first_argument_type &__x, const typename _Predicate::second_argument_type &__y) const

Protected Attributes

- _Predicate **_M_pred**

6.278.1 Detailed Description

template<typename _Predicate>
class std::binary_negate<_Predicate>

One of the [negation functors](#).

6.278.2 Member Typedef Documentation

first_argument_type

typedef _Predicate::first_argument_type [std::binary_function](#)<_Predicate::first_argument_type , ↵
 _Predicate::second_argument_type , bool >::first_argument_type [inherited]
 first_argument_type is the type of the first argument

result_type

typedef bool [std::binary_function](#)<_Predicate::first_argument_type , _Predicate::second_argument↵
 _type , bool >::result_type [inherited]
 result_type is the return type

second_argument_type

typedef _Predicate::second_argument_type [std::binary_function](#)<_Predicate::first_argument_type ,
 _Predicate::second_argument_type , bool >::second_argument_type [inherited]

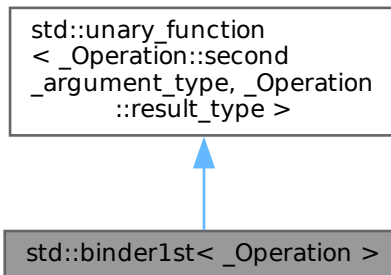
`second_argument_type` is the type of the second argument
 The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.279 std::binder1st<_Operation> Class Template Reference

```
#include <binders.h>
```

Inheritance diagram for `std::binder1st<_Operation>`:



Public Types

- typedef `_Operation::second_argument_type` [argument_type](#)
- typedef `_Operation::result_type` [result_type](#)

Public Member Functions

- **binder1st** (const `_Operation` &__x, const typename `_Operation::first_argument_type` &__y)
- `_Operation::result_type` **operator()** (const typename `_Operation::second_argument_type` &__x) const
- `_Operation::result_type` **operator()** (typename `_Operation::second_argument_type` &__x) const

Protected Attributes

- `_Operation` **op**
- `_Operation::first_argument_type` **value**

6.279.1 Detailed Description

```
template<typename _Operation>
class std::binder1st<_Operation>
```

One of the [binder functors](#).

6.279.2 Member Typedef Documentation

`argument_type`

```
typedef _Operation::second_argument_type std::unary\_function<_Operation::second_argument_type ,
_Operation::result_type >::argument_type [inherited]
```

`argument_type` is the type of the argument

result_type

```
typedef _Operation::result_type std::unary_function< _Operation::second_argument_type , _Operation←
::result_type >::result_type [inherited]
```

result_type is the return type

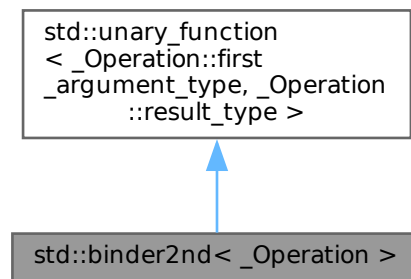
The documentation for this class was generated from the following file:

- [binders.h](#)

6.280 std::binder2nd<_Operation> Class Template Reference

```
#include <binders.h>
```

Inheritance diagram for std::binder2nd<_Operation>:

**Public Types**

- typedef _Operation::first_argument_type [argument_type](#)
- typedef _Operation::result_type [result_type](#)

Public Member Functions

- **binder2nd** (const _Operation &__x, const typename _Operation::second_argument_type &__y)
- _Operation::result_type **operator()** (const typename _Operation::first_argument_type &__x) const
- _Operation::result_type **operator()** (typename _Operation::first_argument_type &__x) const

Protected Attributes

- _Operation **op**
- _Operation::second_argument_type **value**

6.280.1 Detailed Description

```
template<typename _Operation>
```

```
class std::binder2nd<_Operation>
```

One of the [binder functors](#).

6.280.2 Member Typedef Documentation

argument_type

```
typedef _Operation::first_argument_type std::unary_function< _Operation::first_argument_type , _↵  
Operation::result_type >::argument_type [inherited]  
argument_type is the type of the argument
```

result_type

```
typedef _Operation::result_type std::unary_function< _Operation::first_argument_type , _Operation↵  
::result_type >::result_type [inherited]  
result_type is the return type
```

The documentation for this class was generated from the following file:

- [binders.h](#)

6.281 std::binomial_distribution< _IntType > Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _IntType [result_type](#)

Public Member Functions

- **binomial_distribution** (_IntType __t, double __p=0.5)
- **binomial_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- double **p** () const
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()
- _IntType **t** () const

Friends

- template<typename _IntType1, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::binomial_distribution< _IntType1 > &__x)
- bool operator== (const binomial_distribution &__d1, const binomial_distribution &__d2)
- template<typename _IntType1, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::binomial_distribution< _IntType1 > &__x)

6.281.1 Detailed Description

template<typename _IntType = int>
class std::binomial_distribution< _IntType >

A discrete binomial random number distribution.

The formula for the binomial probability density function is $p(i|t, p) = \binom{t}{i} p^i (1-p)^{t-i}$ where t and p are the parameters of the distribution.

6.281.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::binomial_distribution< _IntType >::result_type
```

The type of the range of the distribution.

6.281.3 Member Function Documentation

max()

```
template<typename _IntType = int>
result_type std::binomial_distribution< _IntType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.

min()

```
template<typename _IntType = int>
result_type std::binomial_distribution< _IntType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>() [1/2]

```
template<typename _IntType = int>
template<typename _UniformRandomNumberGenerator >
result_type std::binomial_distribution< _IntType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::binomial_distribution< _IntType >::operator\(\)\(\)](#).

Referenced by [std::binomial_distribution< _IntType >::operator\(\)\(\)](#).

operator>() [2/2]

```
template<typename _IntType >
template<typename _UniformRandomNumberGenerator >
binomial_distribution< _IntType >::result_type std::binomial_distribution< _IntType >::operator()
(
```



```

    _UniformRandomNumberGenerator & __urng,
    const param_type & __param )

```

A rejection algorithm when $t * p \geq 8$ and a simple waiting time method - the second in the referenced book - otherwise. NB: The former is available only if `_GLIBCXX_USE_C99_MATH_TR1` is defined.

Reference: Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. X, Sect. 4 (+ Errata!).

References `std::abs()`, `std::numeric_limits<_Tp>::epsilon()`, `std::log()`, and `std::numeric_limits<_Tp>::max()`.

p()

```

template<typename _IntType = int>
double std::binomial_distribution< _IntType >::p ( ) const [inline]

```

Returns the distribution p parameter.

param() [1/2]

```

template<typename _IntType = int>
param_type std::binomial_distribution< _IntType >::param ( ) const [inline]

```

Returns the parameter set of the distribution.

param() [2/2]

```

template<typename _IntType = int>
void std::binomial_distribution< _IntType >::param (
    const param_type & __param ) [inline]

```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```

template<typename _IntType = int>
void std::binomial_distribution< _IntType >::reset ( ) [inline]

```

Resets the distribution state.

References `std::normal_distribution<_RealType>::reset()`.

t()

```

template<typename _IntType = int>
_IntType std::binomial_distribution< _IntType >::t ( ) const [inline]

```

Returns the distribution t parameter.

6.281.4 Friends And Related Symbol Documentation

operator<<

```

template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::binomial_distribution< _IntType1 > & __x ) [friend]

```

Inserts a binomial_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>binomial_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _IntType = int>
bool operator== (
    const binomial\_distribution< _IntType > & __d1,
    const binomial\_distribution< _IntType > & __d2 ) [friend]
```

Return true if two binomial distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic\_istream< _CharT, _Traits > & operator>> (
    std::basic\_istream< _CharT, _Traits > & __is,
    std::binomial\_distribution< _IntType1 > & __x ) [friend]
```

Extracts a `binomial_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>binomial_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

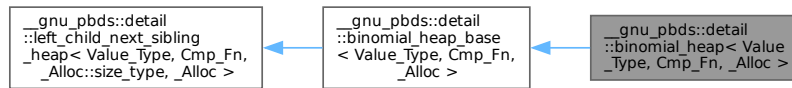
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.282 `__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

```
#include <binomial_heap_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `base_type::allocator_type` **allocator_type**
- typedef `base_type::cmp_fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `base_type::const_pointer` **const_pointer**
- typedef `base_type::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `base_type::pointer` **pointer**
- typedef `base_type::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **binomial_heap** (const `binomial_heap` &)
- **binomial_heap** (const `Cmp_Fn` &)
- **iterator begin** ()
- **const_iterator begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator end** ()
- **const_iterator end** () const
- void **erase** (`point_iterator`)
- template<typename `Pred` >
size_type **erase_if** (`Pred`)
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- void **join** (`binomial_heap_base`< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- size_type **max_size** () const
- void **modify** (`point_iterator`, const_reference)
- void **pop** ()
- `point_iterator` **push** (const_reference)
- size_type **size** () const
- template<typename `Pred` >
void **split** (`Pred`, `binomial_heap_base`< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- void **swap** (`left_child_next_sibling_heap`< `Value_Type`, `Cmp_Fn`, `_Alloc::size_type`, `_Alloc` > &)
- const_reference **top** () const

Protected Types

- typedef `base_type::node` **node**
- typedef `alloc_traits::allocator_type` **node_allocator**
- typedef `_Alloc::size_type` **node_metadata**
- typedef `std::pair< node_pointer, node_pointer >` **node_pointer_pair**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- void **find_max** ()
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap** (`binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** (`left_child_next_sibling_heap` &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_root**
- size_type **m_size**

6.282.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >
```

Binomial heap.

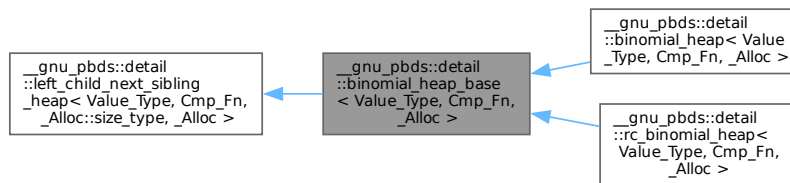
The documentation for this class was generated from the following file:

- [binomial_heap.hpp](#)

6.283 `__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

```
#include <binomial_heap_base_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_v::const_pointer` **const_pointer**
- typedef `__rebind_v::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_v::pointer` **pointer**
- typedef `__rebind_v::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **iterator** **begin** ()
- **const_iterator** **begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator** **end** ()
- **const_iterator** **end** () const
- void **erase** (**point_iterator**)
- template<typename Pred >
size_type **erase_if** (Pred)
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- void **join** (**binomial_heap_base**< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- size_type **max_size** () const
- void **modify** (**point_iterator**, const_reference)
- void **pop** ()
- **point_iterator** **push** (const_reference)
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, **binomial_heap_base**< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- void **swap** (**left_child_next_sibling_heap**< `Value_Type`, `Cmp_Fn`, `_Alloc::size_type`, `_Alloc` > &)
- const_reference **top** () const

Protected Types

- typedef `base_type::node` **node**
- typedef `alloc_traits::allocator_type` **node_allocator**
- typedef `base_type::node_const_pointer` **node_const_pointer**
- typedef `_Alloc::size_type` **node_metadata**
- typedef `base_type::node_pointer` **node_pointer**
- typedef `std::pair< node_pointer, node_pointer >` **node_pointer_pair**

Protected Member Functions

- **binomial_heap_base** (const `binomial_heap_base`< Value_Type, Cmp_Fn, _Alloc > &)
- **binomial_heap_base** (const Cmp_Fn &)
- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- void **find_max** ()
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap** (`binomial_heap_base`< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** (`left_child_next_sibling_heap` &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_root**
- size_type **m_size**

6.283.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >
```

Base class for binomial heap.

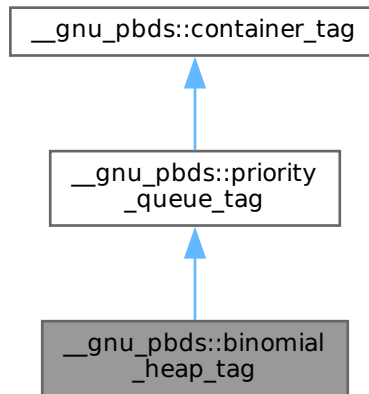
The documentation for this class was generated from the following file:

- [binomial_heap_base.hpp](#)

6.284 `__gnu_pbds::binomial_heap_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::binomial_heap_tag`:



6.284.1 Detailed Description

Binomial-heap.

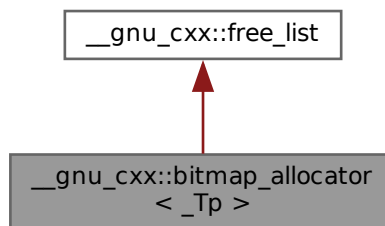
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.285 `__gnu_cxx::bitmap_allocator<_Tp>` Class Template Reference

```
#include <bitmap_allocator.h>
```

Inheritance diagram for `__gnu_cxx::bitmap_allocator<_Tp>`:



Public Types

- `typedef free_list::__mutex_type __mutex_type`
- `typedef const _Tp * const_pointer`

- typedef const _Tp & **const_reference**
- typedef std::ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **bitmap_allocator** (const [bitmap_allocator](#) &) noexcept
- template<typename _Tp1 >
 bitmap_allocator (const [bitmap_allocator](#)<_Tp1 > &) noexcept
- pointer [_M_allocate_single_object](#) ()
- void [_M_deallocate_single_object](#) (pointer __p) throw ()
- const_pointer **address** (const_reference __r) const noexcept
- pointer **address** (reference __r) const noexcept
- pointer **allocate** (size_type __n)
- pointer **allocate** (size_type __n, typename [bitmap_allocator](#)< void >::const_pointer)
- template<typename _Up, typename... _Args>
 void **construct** (_Up * __p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n) throw ()
- template<typename _Up >
 void **destroy** (_Up * __p)
- size_type **max_size** () const noexcept

6.285.1 Detailed Description

template<typename _Tp>
class `__gnu_cxx::bitmap_allocator<_Tp>`

Bitmap Allocator, primary template.

6.285.2 Member Function Documentation

`_M_allocate_single_object()`

```
template<typename _Tp >
pointer __gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object ( ) [inline]
Allocates memory for a single object of size sizeof(_Tp).
```

Exceptions

<code>std::bad_alloc.</code>	If memory cannot be allocated.
------------------------------	--------------------------------

Complexity: Worst case complexity is O(N), but that is hardly ever hit. If and when this particular case is encountered, the next few cases are guaranteed to have a worst case complexity of O(1)! That's why this function performs very well on average. You can consider this function to have a complexity referred to commonly as: Amortized Constant time.

References [__gnu_cxx::__detail::__bit_allocate\(\)](#), [__gnu_cxx::__detail::__num_bitmaps\(\)](#), and [__gnu_cxx::_Bit_scan_forward\(\)](#).

`_M_deallocate_single_object()`

```
template<typename _Tp >
void __gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object (
```



```
pointer __p ) throw ( ) [inline]
```

Deallocates memory that belongs to a single object of size `sizeof(_Tp)`.

Complexity: $O(\lg(N))$, but the worst case is not hit often! This is because containers usually deallocate memory close to each other and this case is handled in $O(1)$ time by the `deallocate` function.

References [__gnu_cxx::__detail::__bit_free\(\)](#), [__gnu_cxx::__detail::__num_bitmaps\(\)](#), and [__gnu_cxx::free_list::_M_insert\(\)](#).

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

6.286 `std::__debug::bitset<_Nb>` Class Template Reference

Inherits `bitset<_Nb>`.

Public Types

- typedef `_Base::reference` **reference**

Public Member Functions

- **bitset** (const [_Base](#) &__x)
- template<typename `_CharT` >
bitset (const `_CharT` *__str, typename [std::basic_string](#)< `_CharT` >::size_type __n=[std::basic_string](#)< `_CharT` >::npos, `_CharT` __zero=`_CharT`('0'), `_CharT` __one=`_CharT`('1'))
- template<class `_CharT`, class `_Traits`, class `_Alloc` >
bitset (const [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` > &__str, typename [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` >::size_type __pos, typename [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` >::size_type __n, `_CharT` __zero, `_CharT` __one=`_CharT`('1'))
- template<typename `_CharT`, typename `_Traits`, typename `_Alloc` >
bitset (const [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` > &__str, typename [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` >::size_type __pos=0, typename [std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` >::size_type __n=([std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` >::npos))
- constexpr **bitset** (unsigned long long __val) noexcept
- const [_Base](#) & **_M_base** () const noexcept
- [_Base](#) & **_M_base** () noexcept
- **bitset**< `_Nb` > & **flip** () noexcept
- **bitset**< `_Nb` > & **flip** (size_t __pos)
- **bitset**< `_Nb` > & **operator&=** (const **bitset**< `_Nb` > &__rhs) noexcept
- **bitset**< `_Nb` > **operator<<** (size_t __pos) const noexcept
- **bitset**< `_Nb` > & **operator<<=** (size_t __pos) noexcept
- bool **operator==** (const **bitset**< `_Nb` > &__rhs) const noexcept
- **bitset**< `_Nb` > **operator>>** (size_t __pos) const noexcept
- **bitset**< `_Nb` > & **operator>>=** (size_t __pos) noexcept
- reference **operator[]** (size_t __pos)
- constexpr bool **operator[]** (size_t __pos) const
- **bitset**< `_Nb` > & **operator^=** (const **bitset**< `_Nb` > &__rhs) noexcept
- **bitset**< `_Nb` > & **operator|=** (const **bitset**< `_Nb` > &__rhs) noexcept
- **bitset**< `_Nb` > **operator~** () const noexcept
- **bitset**< `_Nb` > & **reset** () noexcept
- **bitset**< `_Nb` > & **reset** (size_t __pos)
- **bitset**< `_Nb` > & **set** () noexcept
- **bitset**< `_Nb` > & **set** (size_t __pos, bool __val=true)
- template<typename `_CharT`, typename `_Traits`, typename `_Alloc` >
[std::basic_string](#)< `_CharT`, `_Traits`, `_Alloc` > **to_string** () const

- `template<typename _CharT, typename _Traits >`
`std::basic_string< _CharT, _Traits, std::allocator< _CharT > > to_string () const`
- `template<typename _CharT >`
`std::basic_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT > > to_string () const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string () const`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_string< _CharT, _Traits, _Alloc > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `template<class _CharT, class _Traits >`
`std::basic_string< _CharT, _Traits, std::allocator< _CharT > > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `template<class _CharT >`
`std::basic_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT > > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string (char __zero, char __one='1') const`

6.286.1 Detailed Description

`template<size_t _Nb>`

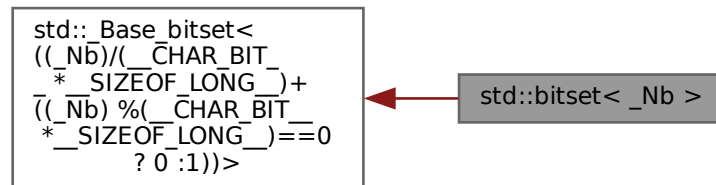
`class std::__debug::bitset< _Nb >`

Class std::bitset with additional safety/checking/debug instrumentation.
The documentation for this class was generated from the following file:

- [debug/bitset](#)

6.287 std::bitset< _Nb > Class Template Reference

Inheritance diagram for std::bitset< _Nb >:



Classes

- class [reference](#)

Public Member Functions

- constexpr `bitset ()` noexcept
- `template<typename _CharT >`
`bitset (const _CharT * __str, typename std::basic_string< _CharT >::size_type __n=std::basic_string< _CharT >::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))`

- `template<class _CharT, class _Traits, class _Alloc >`
`bitset (const std::basic_string< _CharT, _Traits, _Alloc > &__s, size_t __position, size_t __n)`
- `template<class _CharT, class _Traits, class _Alloc >`
`bitset (const std::basic_string< _CharT, _Traits, _Alloc > &__s, size_t __position, size_t __n, _CharT __zero, _CharT __one=_CharT('1'))`
- `template<class _CharT, class _Traits, class _Alloc >`
`bitset (const std::basic_string< _CharT, _Traits, _Alloc > &__s, size_t __position=0)`
- `constexpr bitset (unsigned long long __val) noexcept`
- `size_t Find_first () const noexcept`
- `size_t Find_next (size_t __prev) const noexcept`
- `template<class _CharT, class _Traits >`
`void M_copy_from_ptr (const _CharT *, size_t, size_t, size_t, _CharT, _CharT)`
- `template<class _CharT, class _Traits, class _Alloc >`
`void M_copy_from_string (const std::basic_string< _CharT, _Traits, _Alloc > &__s, size_t __pos, size_t __n)`
- `template<class _CharT, class _Traits, class _Alloc >`
`void M_copy_from_string (const std::basic_string< _CharT, _Traits, _Alloc > &__s, size_t __pos, size_t __n, _CharT __zero, _CharT __one)`
- `template<class _CharT, class _Traits, class _Alloc >`
`void M_copy_to_string (std::basic_string< _CharT, _Traits, _Alloc > &, _CharT, _CharT) const`
- `template<class _CharT, class _Traits, class _Alloc >`
`void M_copy_to_string (std::basic_string< _CharT, _Traits, _Alloc > &__s) const`
- `bool all () const noexcept`
- `bool any () const noexcept`
- `size_t count () const noexcept`
- `bitset< _Nb > & flip () noexcept`
- `bitset< _Nb > & flip (size_t __position)`
- `bool none () const noexcept`
- `bitset< _Nb > operator~ () const noexcept`
- `bitset< _Nb > & reset () noexcept`
- `bitset< _Nb > & reset (size_t __position)`
- `bitset< _Nb > & set () noexcept`
- `bitset< _Nb > & set (size_t __position, bool __val=true)`
- `constexpr size_t size () const noexcept`
- `bool test (size_t __position) const`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_string< _CharT, _Traits, _Alloc > to_string () const`
- `template<class _CharT, class _Traits >`
`std::basic_string< _CharT, _Traits, std::allocator< _CharT > > to_string () const`
- `template<class _CharT >`
`std::basic_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT > > to_string () const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string () const`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_string< _CharT, _Traits, _Alloc > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `template<class _CharT, class _Traits >`
`std::basic_string< _CharT, _Traits, std::allocator< _CharT > > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `template<class _CharT >`
`std::basic_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT > > to_string (_CharT __zero, _CharT __one=_CharT('1')) const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string (char __zero, char __one='1') const`
- `unsigned long long to_ullong () const`

- unsigned long [to_ulong](#) () const
- [bitset<_Nb> & operator&=](#) (const [bitset<_Nb> &__rhs](#)) noexcept
- [bitset<_Nb> & operator|=](#) (const [bitset<_Nb> &__rhs](#)) noexcept
- [bitset<_Nb> & operator^=](#) (const [bitset<_Nb> &__rhs](#)) noexcept
- [bitset<_Nb> & operator<<=](#) (size_t __position) noexcept
- [bitset<_Nb> & operator>>=](#) (size_t __position) noexcept
- [bitset<_Nb> & _Unchecked_set](#) (size_t __pos) noexcept
- [bitset<_Nb> & _Unchecked_set](#) (size_t __pos, int __val) noexcept
- [bitset<_Nb> & _Unchecked_reset](#) (size_t __pos) noexcept
- [bitset<_Nb> & _Unchecked_flip](#) (size_t __pos) noexcept
- constexpr bool [_Unchecked_test](#) (size_t __pos) const noexcept
- [reference operator\[\]](#) (size_t __position)
- constexpr bool [operator\[\]](#) (size_t __position) const
- bool [operator==](#) (const [bitset<_Nb> &__rhs](#)) const noexcept
- [bitset<_Nb> operator<<](#) (size_t __position) const noexcept
- [bitset<_Nb> operator>>](#) (size_t __position) const noexcept

Friends

- class **reference**
- struct **std::hash<bitset>**

6.287.1 Detailed Description

template<size_t _Nb>
class std::bitset<_Nb>

The `bitset` class represents a *fixed-size* sequence of bits.

(Note that `bitset` does *not* meet the formal requirements of a [container](#). Mainly, it lacks iterators.)

The template argument, *Nb*, may be any non-negative number, specifying the number of bits (e.g., "0", "12", "1024*1024").

In the general unoptimized case, storage is allocated in word-sized blocks. Let *B* be the number of bits in a word, then $(Nb+(B-1))/B$ words will be used for storage. $B - Nb \bmod B$ bits are unused. (They are the high-order bits in the highest word.) It is a class invariant that those unused bits are always zero.

If you think of `bitset` as a *simple array of bits*, be aware that your mental picture is reversed: a `bitset` behaves the same way as bits in integers do, with the bit at index 0 in the *least significant* / *right-hand* position, and the bit at index *Nb*-1 in the *most significant* / *left-hand* position. Thus, unlike other containers, a `bitset`'s index *counts from right to left*, to put it very loosely.

This behavior is preserved when translating to and from strings. For example, the first line of the following program probably prints *b('a')* is *0001100001* on a modern ASCII system.

```

#include <bitset>
#include <iostream>
#include <sstream>

using namespace std;

int main()
{
    long        a = 'a';
    bitset<10>   b(a);

    cout << "b('a') is " << b << endl;

    ostringstream s;
    s << b;
    string str = s.str();
    cout << "index 3 in the string is " << str[3] << " but\n"
         << "index 3 in the bitset is " << b[3] << endl;
}

```

Also see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/ext_containers.html for a description of extensions.

Most of the actual code isn't contained in `bitset<>` itself, but in the base class `_Base_bitset`. The base class works with whole words, not with individual bits. This allows us to specialize `_Base_bitset` for the important special case where the bitset is only a single word.

Extra confusion can result due to the fact that the storage for `_Base_bitset` is a regular array, and is indexed as such. This is carefully encapsulated.

6.287.2 Constructor & Destructor Documentation

`bitset()` [1/5]

```

template<size_t _Nb>
constexpr std::bitset< _Nb >::bitset ( ) [inline], [constexpr], [noexcept]

```

All bits set to zero.

`bitset()` [2/5]

```

template<size_t _Nb>
constexpr std::bitset< _Nb >::bitset (
    unsigned long long __val ) [inline], [constexpr], [noexcept]

```

Initial bits bitwise-copied from a single word (others set to zero).

`bitset()` [3/5]

```

template<size_t _Nb>
template<class _CharT , class _Traits , class _Alloc >
std::bitset< _Nb >::bitset (
    const std::basic_string< _CharT, _Traits, _Alloc > & __s,
    size_t __position = 0 ) [inline], [explicit]

```

Use a subset of a string.

Parameters

<code>__s</code>	A string of 0 and 1 characters.
<code>__position</code>	Index of the first character in <code>__s</code> to use; defaults to zero.

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is bigger the size of <code>__s</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither 0 nor 1.

bitset() [4/5]

```
template<size_t _Nb>
template<class _CharT , class _Traits , class _Alloc >
std::bitset<_Nb>::bitset (
    const std::basic_string<_CharT, _Traits, _Alloc> & __s,
    size_t __position,
    size_t __n ) [inline]
```

Use a subset of a string.

Parameters

<code>__s</code>	A string of 0 and 1 characters.
<code>__position</code>	Index of the first character in <code>__s</code> to use.
<code>__n</code>	The number of characters to copy.

Exceptions

<code>std::out_of_range</code>	If <code>__position</code> is bigger the size of <code>__s</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither 0 nor 1.

bitset() [5/5]

```
template<size_t _Nb>
template<typename _CharT >
std::bitset<_Nb>::bitset (
    const _CharT * __str,
    typename std::basic_string<_CharT>::size_type __n = std::basic_string<_CharT>::npos,
    _CharT __zero = _CharT('0'),
    _CharT __one = _CharT('1') ) [inline], [explicit]
```

Construct from a character array.

Parameters

<code>__str</code>	An array of characters <i>zero</i> and <i>one</i> .
<code>__n</code>	The number of characters to use.
<code>__zero</code>	The character corresponding to the value 0.
<code>__one</code>	The character corresponding to the value 1.

Exceptions

<code>std::invalid_argument</code>	If a character appears in the string which is neither <code>__zero</code> nor <code>__one</code> .
------------------------------------	--

6.287.3 Member Function Documentation**all()**

```
template<size_t _Nb>
```

```
bool std::bitset<_Nb>::all ( ) const [inline], [noexcept]
```

Tests whether all the bits are on.

Returns

True if all the bits are set.

any()

```
template<size_t _Nb>
```

```
bool std::bitset<_Nb>::any ( ) const [inline], [noexcept]
```

Tests whether any of the bits are on.

Returns

True if at least one bit is set.

count()

```
template<size_t _Nb>
```

```
size_t std::bitset<_Nb>::count ( ) const [inline], [noexcept]
```

Returns the number of bits which are set.

flip() [1/2]

```
template<size_t _Nb>
```

```
bitset<_Nb> & std::bitset<_Nb>::flip ( ) [inline], [noexcept]
```

Toggles every bit to its opposite value.

flip() [2/2]

```
template<size_t _Nb>
```

```
bitset<_Nb> & std::bitset<_Nb>::flip (
    size_t __position ) [inline]
```

Toggles a given bit to its opposite value.

Parameters

<code>__position</code>	The index of the bit.
-------------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

none()

```
template<size_t _Nb>
```

```
bool std::bitset<_Nb>::none ( ) const [inline], [noexcept]
```

Tests whether any of the bits are on.

Returns

True if none of the bits are set.

operator&=()

```
template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::operator&= (
    const bitset<_Nb> & __rhs ) [inline], [noexcept]
```

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

operator<<()

```
template<size_t _Nb>
bitset<_Nb> > std::bitset<_Nb>::operator<< (
    size_t __position ) const [inline], [noexcept]
```

Self-explanatory.

operator<<=()

```
template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::operator<<= (
    size_t __position ) [inline], [noexcept]
```

Operations on bitsets.

Parameters

<code>__position</code>	The number of places to shift.
-------------------------	--------------------------------

These should be self-explanatory.

operator==()

```
template<size_t _Nb>
bool std::bitset<_Nb>::operator== (
    const bitset<_Nb> & __rhs ) const [inline], [noexcept]
```

These comparisons for equality/inequality are, well, *bitwise*.

operator>>()

```
template<size_t _Nb>
bitset<_Nb> > std::bitset<_Nb>::operator>> (
    size_t __position ) const [inline], [noexcept]
```

Self-explanatory.

operator>>=()

```
template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::operator>>= (
    size_t __position ) [inline], [noexcept]
```

Operations on bitsets.

Parameters

<code>__position</code>	The number of places to shift.
-------------------------	--------------------------------

These should be self-explanatory.

operator[]() [1/2]

```
template<size_t _Nb>
reference std::bitset< _Nb >::operator[] (
    size_t __position ) [inline]
```

Array-indexing support.

Parameters

<code>__position</code>	Index into the bitset.
-------------------------	------------------------

Returns

A bool for a *const bitset*. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

_GLIBCXX_RESOLVE_LIB_DEFECTS Note that this implementation already resolves DR 11 (items 1 and 2), but does not do the range-checking required by that DR's resolution. -pme The DR has since been changed: range-checking is a precondition (users' responsibility), and these functions must not throw. -pme

operator[]() [2/2]

```
template<size_t _Nb>
constexpr bool std::bitset< _Nb >::operator[] (
    size_t __position ) const [inline], [constexpr]
```

Array-indexing support.

Parameters

<code>__position</code>	Index into the bitset.
-------------------------	------------------------

Returns

A bool for a *const bitset*. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

_GLIBCXX_RESOLVE_LIB_DEFECTS Note that this implementation already resolves DR 11 (items 1 and 2), but does not do the range-checking required by that DR's resolution. -pme The DR has since been changed: range-checking is a precondition (users' responsibility), and these functions must not throw. -pme

operator^=()

```
template<size_t _Nb>
```

```

bitset<_Nb> & std::bitset<_Nb>::operator^= (
    const bitset<_Nb> & __rhs ) [inline], [noexcept]

```

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

`operator" |=()`

```

template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::operator|= (
    const bitset<_Nb> & __rhs ) [inline], [noexcept]

```

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

`operator~()`

```

template<size_t _Nb>
bitset<_Nb> std::bitset<_Nb>::operator~ ( ) const [inline], [noexcept]

```

See the no-argument `flip()`.

`reset()` [1/2]

```

template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::reset ( ) [inline], [noexcept]

```

Sets every bit to false.

`reset()` [2/2]

```

template<size_t _Nb>
bitset<_Nb> & std::bitset<_Nb>::reset (
    size_t __position ) [inline]

```

Sets a given bit to false.

Parameters

<code>__position</code>	The index of the bit.
-------------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is bigger the size of the set.
--------------------------------	--

Same as writing `set(pos, false)`.

set() [1/2]

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::set ( ) [inline], [noexcept]
```

Sets every bit to true.

set() [2/2]

```
template<size_t _Nb>
bitset< _Nb > & std::bitset< _Nb >::set (
    size_t __position,
    bool __val = true ) [inline]
```

Sets a given bit to a particular value.

Parameters

<code>__position</code>	The index of the bit.
<code>__val</code>	Either true or false, defaults to true.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

size()

```
template<size_t _Nb>
constexpr size_t std::bitset< _Nb >::size ( ) const [inline], [constexpr], [noexcept]
```

Returns the total number of bits.

test()

```
template<size_t _Nb>
bool std::bitset< _Nb >::test (
    size_t __position ) const [inline]
```

Tests the value of a bit.

Parameters

<code>__position</code>	The index of a bit.
-------------------------	---------------------

Returns

The value at *pos*.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

to_string()

```
template<size_t _Nb>
template<class _CharT , class _Traits , class _Alloc >
std::basic_string< _CharT, _Traits, _Alloc > std::bitset< _Nb >::to_string ( ) const [inline]
```

Returns a character interpretation of the bitset.

Returns

The string equivalent of the bits.

Note the ordering of the bits: decreasing character positions correspond to increasing bit positions (see the main class notes for an example).

to_ulong()

```
template<size_t _Nb>
unsigned long std::bitset< _Nb >::to_ulong ( ) const [inline]
```

Returns a numerical interpretation of the bitset.

Returns

The integral equivalent of the bits.

Exceptions

<i>std::overflow_error</i>	If there are too many bits to be represented in an unsigned long.
----------------------------	---

The documentation for this class was generated from the following file:

- [bitset](#)

6.288 std::tr2::bool_set Class Reference**Public Member Functions**

- constexpr [bool_set](#) ()
- constexpr [bool_set](#) (bool __t)
- bool **contains** ([bool_set](#) __b) const
- bool **equals** ([bool_set](#) __b) const
- bool **is_emptyset** () const
- bool **is_indeterminate** () const
- bool **is_singleton** () const
- [operator bool](#) () const

Static Public Member Functions

- static [bool_set](#) **emptyset** ()
- static [bool_set](#) **indeterminate** ()

Friends

- [bool_set](#) **operator!** ([bool_set](#) __b)
- [bool_set](#) **operator&** ([bool_set](#) __s, [bool_set](#) __t)
- template<typename CharT , typename Traits >
[std::basic_ostream](#)< CharT, Traits > & **operator<<** ([std::basic_ostream](#)< CharT, Traits > &__out, [bool_set](#) __b)

- `bool_set operator== (bool_set __s, bool_set __t)`
- `template<typename CharT, typename Traits >
std::basic_istream< CharT, Traits > & operator>> (std::basic_istream< CharT, Traits > &__in, bool_set &__b)`
- `bool_set operator^ (bool_set __s, bool_set __t)`
- `bool_set operator| (bool_set __s, bool_set __t)`

6.288.1 Detailed Description

`bool_set`

See N2136, `Bool_set`: multi-valued logic by Hervé Brönnimann, Guillaume Melquiond, Sylvain Pion.

The implicit conversion to `bool` is slippery! I may use the new explicit conversion. This has been specialized in the language so that in contexts requiring a `bool` the conversion happens implicitly. Thus most objections should be eliminated.

6.288.2 Constructor & Destructor Documentation

`bool_set()` [1/2]

```
constexpr std::tr2::bool_set::bool_set ( ) [inline], [constexpr]  
Default constructor.
```

`bool_set()` [2/2]

```
constexpr std::tr2::bool_set::bool_set (  
    bool __t ) [inline], [constexpr]  
Constructor from bool.
```

6.288.3 Member Function Documentation

`equals()`

```
bool std::tr2::bool_set::equals (  
    bool_set __b ) const [inline]  
Return true if states are equal.
```

`is_emptyset()`

```
bool std::tr2::bool_set::is_emptyset ( ) const [inline]  
Return true if this is empty.
```

`is_indeterminate()`

```
bool std::tr2::bool_set::is_indeterminate ( ) const [inline]  
Return true if this is indeterminate.
```

`is_singleton()`

```
bool std::tr2::bool_set::is_singleton ( ) const [inline]  
Return true if this is false or true (normal boolean).
```

`operator bool()`

```
std::tr2::bool_set::operator bool ( ) const [inline]  
Conversion to bool.
```

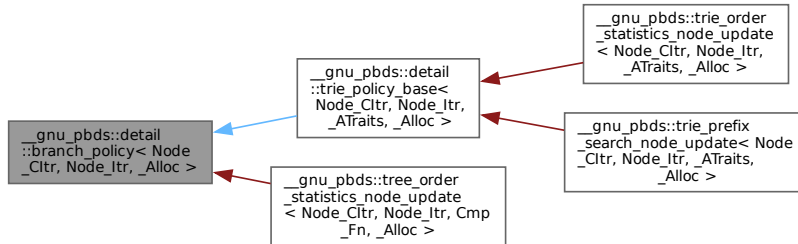
The documentation for this class was generated from the following files:

- `bool_set`
- `bool_set.tcc`

6.289 `__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >` Struct Template Reference

```
#include <branch_policy.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >`:



Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Itr::value_type` **it_type**
- typedef `rebind_k::const_reference` **key_const_reference**
- typedef `value_type::first_type` **key_type**
- typedef `remove_const< key_type >::type` **rkey_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `rebind_traits< _Alloc, rkey_type >` **rebind_k**
- typedef `rebind_traits< _Alloc, rcvalue_type >` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual `it_type end ()=0`
- `it_type end_iterator () const`

Static Protected Member Functions

- static `key_const_reference extract_key (const_reference r_val)`

6.289.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _Alloc>
struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >
```

Primary template, base class for branch structure policies.

The documentation for this struct was generated from the following file:

- [branch_policy.hpp](#)

6.290 `__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >` Struct Template Reference

```
#include <branch_policy.hpp>
```

Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Cltr::value_type` **it_type**
- typedef `rebind_v::const_reference` **key_const_reference**
- typedef `value_type` **key_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `rebind_traits< _Alloc, rcvalue_type >` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual `it_type` **end** () const =0
- `it_type` **end_iterator** () const

Static Protected Member Functions

- static `key_const_reference` **extract_key** (const_reference r_val)

6.290.1 Detailed Description

```
template<typename Node_Cltr, typename _Alloc>
struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >
```

Specialization for const iterators.

The documentation for this struct was generated from the following file:

- [branch_policy.hpp](#)

6.291 `std::cauchy_distribution< _RealType >` Class Template Reference

```
#include <random.h>
```

Classes

- struct `param_type`

Public Types

- typedef `_RealType` **result_type**

Public Member Functions

- **cauchy_distribution** (`_RealType` __a, `_RealType` __b=1.0)
- **cauchy_distribution** (const `param_type` &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng)

- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator>`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `template<typename _UniformRandomNumberGenerator>`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `_RealType a () const`
- `_RealType b () const`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `param_type param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `bool operator== (const cauchy_distribution &__d1, const cauchy_distribution &__d2)`

6.291.1 Detailed Description

`template<typename _RealType = double>`
class `std::cauchy_distribution<_RealType>`

A `cauchy_distribution` random number distribution.

The formula for the normal probability mass function is $p(x|a, b) = (\pi b(1 + (\frac{x-a}{b})^2))^{-1}$

6.291.2 Member Typedef Documentation

result_type

`template<typename _RealType = double>`
`typedef _RealType std::cauchy_distribution<_RealType>::result_type`
 The type of the range of the distribution.

6.291.3 Member Function Documentation

max()

`template<typename _RealType = double>`
`result_type std::cauchy_distribution<_RealType>::max () const [inline]`
 Returns the least upper bound value of the distribution.
 References `std::numeric_limits<_Tp>::max()`.

min()

`template<typename _RealType = double>`
`result_type std::cauchy_distribution<_RealType>::min () const [inline]`
 Returns the greatest lower bound value of the distribution.
 References `std::numeric_limits<_Tp>::lowest()`.

operator()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::cauchy_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::cauchy_distribution< _RealType >::operator\(\)\(\)](#).

Referenced by [std::cauchy_distribution< _RealType >::operator\(\)\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::cauchy_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

Referenced by [std::operator>>\(\)](#).

param() [2/2]

```
template<typename _RealType = double>
void std::cauchy_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::cauchy_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

6.291.4 Friends And Related Symbol Documentation**operator==**

```
template<typename _RealType = double>
bool operator== (
    const cauchy_distribution< _RealType > & __d1,
    const cauchy_distribution< _RealType > & __d2 ) [friend]
```

Return true if two Cauchy distributions have the same parameters.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.292 `__gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >` Class Template Reference

```
#include <hash_policy.hpp>
```

Public Types

- enum { [external_load_access](#) }
- typedef `Size_Type` **size_type**

Public Member Functions

- [cc_hash_max_collision_check_resize_trigger](#) (float load=0.5)
- float [get_load](#) () const
- void [set_load](#) (float load)
- void **swap** ([cc_hash_max_collision_check_resize_trigger](#)< `External_Load_Access`, `Size_Type` > &other)

Protected Member Functions

- bool [is_grow_needed](#) (size_type size, size_type num_entries) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) (size_type num_entries)
- void [notify_externally_resized](#) (size_type new_size)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) (size_type num_entries)
- void [notify_resized](#) (size_type new_size)

6.292.1 Detailed Description

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >
```

A resize trigger policy based on collision checks. It keeps the simulated load factor lower than some given load factor.

6.292.2 Member Enumeration Documentation

anonymous enum

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
anonymous enum
```

Enumerator

<code>external_load_access</code>	Specifies whether the load factor can be accessed externally. The two options have different trade-offs in terms of flexibility, genericity, and encapsulation.
-----------------------------------	---

6.292.3 Constructor & Destructor Documentation

cc_hash_max_collision_check_resize_trigger()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
__gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::cc_↔
hash_max_collision_check_resize_trigger (
    float load = 0.5 )
```

Default constructor, or constructor taking load, a __load factor which it will attempt to maintain.

6.292.4 Member Function Documentation

get_load()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
float __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::get_load ( ) const [inline]
```

Returns the current load.

is_grow_needed()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
bool __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::is_grow_needed (
    size_type size,
    size_type num_entries ) const [inline], [protected]
```

Queries whether a grow is needed. This method is called only if this object indicated is needed.

is_resize_needed()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
bool __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::is_resize_needed ( ) const [inline], [protected]
```

Queries whether a resize is needed.

notify_cleared()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_cleared ( ) [protected]
```

Notifies the table was cleared.

notify_erase_search_collision()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_erase_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_erase_search_end()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_erase_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_erase_search_start()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_erase_search_start ( ) [inline], [protected]
```

Notifies an erase search started.

notify_erased()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_erased (
    size_type num_entries ) [inline], [protected]
```

Notifies an element was erased.

notify_externally_resized()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_externally_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized externally.

notify_find_search_collision()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_find_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_find_search_end()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_find_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_find_search_start()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_find_search_start ( ) [inline], [protected]
```

Notifies a find search started.

notify_insert_search_collision()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_insert_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_insert_search_end()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↵
::notify_insert_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_insert_search_start()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_insert_search_start ( ) [inline], [protected]
```

Notifies an insert search started.

notify_inserted()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_inserted (
    size_type num_entries ) [inline], [protected]
```

Notifies an element was inserted.

notify_resized()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::notify_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized as a result of this object's signifying that a resize is needed.

set_load()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >↔
::set_load (
    float load )
```

Sets the load; does not resize the container.

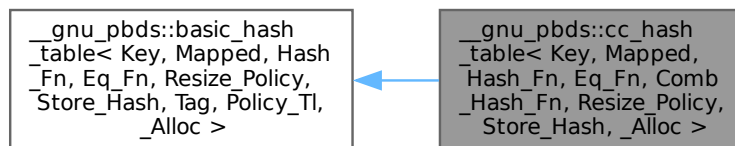
The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.293 __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc > Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >`:



Public Types

- typedef Comb_Hash_Fn **comb_hash_fn**
- typedef [cc_hash_tag](#) **container_category**
- typedef Eq_Fn **eq_fn**
- typedef Hash_Fn **hash_fn**
- typedef Resize_Policy **resize_policy**

Public Member Functions

- [cc_hash_table](#) ()
- **cc_hash_table** (const [cc_hash_table](#) &other)
- [cc_hash_table](#) (const hash_fn &h)
- [cc_hash_table](#) (const hash_fn &h, const eq_fn &e)
- [cc_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch)
- [cc_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch, const resize_policy &rp)
- template<typename It >
[cc_hash_table](#) (It first, It last)
- template<typename It >
[cc_hash_table](#) (It first, It last, const hash_fn &h)
- template<typename It >
[cc_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e)
- template<typename It >
[cc_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch)
- template<typename It >
[cc_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch, const resize_policy &rp)
- [cc_hash_table](#) & **operator=** (const [cc_hash_table](#) &other)
- void **swap** ([cc_hash_table](#) &other)

6.293.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash,
typename _Alloc = std::allocator<char>>>
class __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >
```

A collision-chaining hash-based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Comb_Hash_Fn</i>	Combining hash functor. If Hash_Fn is not null_type, then this is the ranged-hash functor; otherwise, this is the range-hashing functor. XXX(See Design::Hash-Based Containers::Hash Policies.)
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key. If Hash_Fn is null_type, then the container will not compile if this value is true
<i>_Alloc</i>	Allocator type.

Base tag choices are: `cc_hash_tag`.

Base is `basic_hash_table`.

6.293.2 Constructor & Destructor Documentation

`cc_hash_table()` [1/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_↵
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std↵
::allocator<char>>>
```

```
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table ( ) [inline]
```

Default constructor.

`cc_hash_table()` [2/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_↵
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std↵
::allocator<char>>>
```

```
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    const hash_fn & h ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `Hash_Fn` object of the container object.

`cc_hash_table()` [3/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_↵
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std↵
::allocator<char>>>
```

```
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    const hash_fn & h,
    const eq_fn & e ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

`cc_hash_table()` [4/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_↵
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std↵
::allocator<char>>>
```

```
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    const hash_fn & h,
    const eq_fn & e,
    const comb_hash_fn & ch ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object.

`cc_hash_table()` [5/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (
    const hash_fn & h,
    const eq_fn & e,
    const comb_hash_fn & ch,
    const resize_policy & rp ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object, and `r_resize_policy` will be copied by the `resize_policy` object of the container object.

`cc_hash_table()` [6/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (
    It first,
    It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

`cc_hash_table()` [7/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (
    It first,
    It last,
    const hash_fn & h ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

`cc_hash_table()` [8/10]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
```



```

Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::
allocator<char>>
template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e ) [inline]

```

Constructor taking __iterators to a range of value_types and some policy objects The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, and r_eq_fn will be copied by the eq_fn object of the container object.

cc_hash_table() [9/10]

```

template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::
allocator<char>>
template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e,
    const comb_hash_fn & ch ) [inline]

```

Constructor taking __iterators to a range of value_types and some policy objects The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, and r_comb_hash_fn will be copied by the comb_hash_fn object of the container object.

cc_hash_table() [10/10]

```

template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn =
detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_
policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::
allocator<char>>
template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e,
    const comb_hash_fn & ch,
    const resize_policy & rp ) [inline]

```

Constructor taking __iterators to a range of value_types and some policy objects The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, r_comb_hash_fn will be copied by the comb_hash_fn

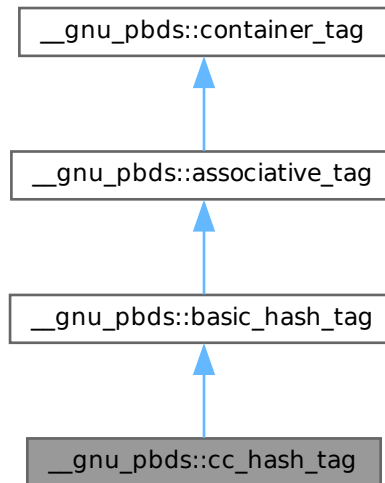
object of the container object, and `r_resize_policy` will be copied by the `resize_policy` object of the container object. The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.294 `__gnu_pbds::cc_hash_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::cc_hash_tag`:



6.294.1 Detailed Description

Collision-chaining hash.

The documentation for this struct was generated from the following file:

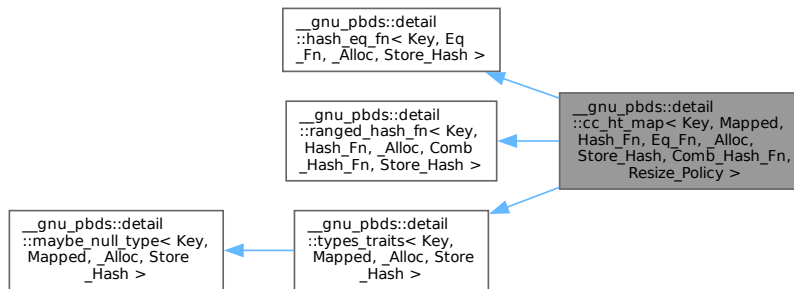
- [tag_and_trait.hpp](#)

6.295 `__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >` Class Template Reference

```
#include <cc_ht_map_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_↵`

_Hash_Fn, Resize_Policy >:



Public Types

- enum { **store_hash** }
- typedef _Alloc **allocator_type**
- typedef Comb_Hash_Fn **comb_hash_fn**
- typedef const_iterator **const_iterator**
- typedef traits_base::const_pointer **const_pointer**
- typedef traits_base::const_reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef Eq_Fn **eq_fn**
- typedef Hash_Fn **hash_fn**
- typedef iterator **iterator**
- typedef traits_base::key_const_pointer **key_const_pointer**
- typedef traits_base::key_const_reference **key_const_reference**
- typedef traits_base::key_pointer **key_pointer**
- typedef traits_base::key_reference **key_reference**
- typedef traits_base::key_type **key_type**
- typedef traits_base::mapped_const_pointer **mapped_const_pointer**
- typedef traits_base::mapped_const_reference **mapped_const_reference**
- typedef traits_base::mapped_pointer **mapped_pointer**
- typedef traits_base::mapped_reference **mapped_reference**
- typedef traits_base::mapped_type **mapped_type**
- typedef __nothrowcopy::indicator **no_throw_indicator**
- typedef point_const_iterator **point_const_iterator**
- typedef point_iterator **point_iterator**
- typedef traits_base::pointer **pointer**
- typedef traits_base::reference **reference**
- typedef Resize_Policy **resize_policy**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef stored_data< value_type, size_type, Store_Hash > **stored_data_type**
- typedef traits_base::value_type **value_type**

Public Member Functions

- `cc_ht_map` (const `cc_ht_map`< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > &)
- `cc_ht_map` (const Hash_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &, const Comb_Hash_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &, const Comb_Hash_Fn &, const Resize_Policy &)
- iterator `begin` ()
- const_iterator `begin` () const
- void `clear` ()
- template<typename It >
void `copy_from_range` (It, It)
- bool `empty` () const
- iterator `end` ()
- const_iterator `end` () const
- bool `erase` (key_const_reference)
- template<typename Pred >
size_type `erase_if` (Pred)
- point_iterator `find` (key_const_reference)
- point_const_iterator `find` (key_const_reference) const
- point_iterator `find_end` ()
- point_const_iterator `find_end` () const
- Comb_Hash_Fn & `get_comb_hash_fn` ()
- const Comb_Hash_Fn & `get_comb_hash_fn` () const
- Eq_Fn & `get_eq_fn` ()
- const Eq_Fn & `get_eq_fn` () const
- Hash_Fn & `get_hash_fn` ()
- const Hash_Fn & `get_hash_fn` () const
- Resize_Policy & `get_resize_policy` ()
- const Resize_Policy & `get_resize_policy` () const
- void `initialize` ()
- `std::pair`< point_iterator, bool > `insert` (const_reference r_val)
- size_type `max_size` () const
- mapped_reference `operator[]` (key_const_reference r_key)
- size_type `size` () const
- void `swap` (`cc_ht_map`< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > &)

Public Attributes

- no_throw_indicator `m_no_throw_copies_indicator`
- store_extra `m_store_extra_indicator`

Friends

- class `const_iterator_`
- class `iterator_`

6.295.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool
Store_Hash, typename Comb_Hash_Fn, typename Resize_Policy>
class __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn,
Resize_Policy >
```

A collision-chaining hash-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor. Default is <code>__gnu_cxx::hash</code> .
<i>Eq_Fn</i>	Equal functor. Default <code>std::equal_to<Key></code>
<i>_Alloc</i>	Allocator type.
<i>Store_Hash</i>	If key type stores extra metadata. Defaults to false.
<i>Comb_Hash_Fn</i>	Combining hash functor. If <i>Hash_Fn</i> is not <code>null_type</code> , then this is the ranged-hash functor; otherwise, this is the range-hashing functor. XXX(See Design::Hash-Based Containers::Hash Policies.) Default <code>direct_mask_range_hashing</code> .
<i>Resize_Policy</i>	Resizes hash. Defaults to <code>hash_standard_resize_policy</code> , using <code>hash_exponential_size_policy</code> and <code>hash_load_check_resize_trigger</code> .

Bases are: `detail::hash_eq_fn`, `Resize_Policy`, `detail::ranged_hash_fn`, `detail::types_traits`. (Optional: `detail::debug_map_base`.)

6.295.2 Member Enumeration Documentation

anonymous enum

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
```

anonymous enum

Value stores hash, true or false.

6.295.3 Member Function Documentation

empty()

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
```

```
bool __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_
Fn, Resize_Policy >::empty ( ) const [inline]
```

True if `size() == 0`.

get_comb_hash_fn() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
```

```
Comb_Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Hash_Fn, Resize_Policy >::get_comb_hash_fn ( )
```

Return current `comb_hash_fn`.

`get_comb_hash_fn()` [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
const Comb_Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_↵
Hash, Comb_Hash_Fn, Resize_Policy >::get_comb_hash_fn ( ) const
Return current const comb_hash_fn.
```

`get_eq_fn()` [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
Eq_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_↵
Hash_Fn, Resize_Policy >::get_eq_fn ( )
Return current eq_fn.
```

`get_eq_fn()` [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
const Eq_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Hash_Fn, Resize_Policy >::get_eq_fn ( ) const
Return current const eq_fn.
```

`get_hash_fn()` [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_↵
Hash_Fn, Resize_Policy >::get_hash_fn ( )
Return current hash_fn.
```

`get_hash_fn()` [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
const Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Hash_Fn, Resize_Policy >::get_hash_fn ( ) const
Return current const hash_fn.
```

`get_resize_policy()` [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
Resize_Policy & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Hash_Fn, Resize_Policy >::get_resize_policy ( )
Return current resize_policy.
```

`get_resize_policy()` [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy >
const Resize_Policy & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_↵
Hash, Comb_Hash_Fn, Resize_Policy >::get_resize_policy ( ) const
Return current const resize_policy.
```

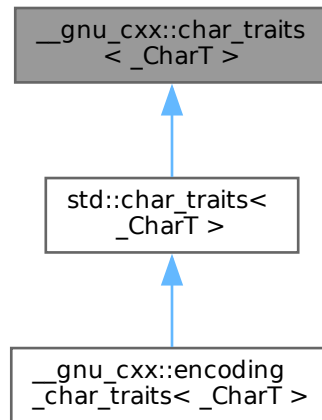
The documentation for this class was generated from the following file:

- [cc_ht_map.hpp](#)

6.296 __gnu_cxx::char_traits<_CharT> Struct Template Reference

```
#include <char_traits.h>
```

Inheritance diagram for __gnu_cxx::char_traits<_CharT>:



Public Types

- typedef `_CharT` **char_type**
- typedef `_Char_types<_CharT>::int_type` **int_type**
- typedef `_Char_types<_CharT>::off_type` **off_type**
- typedef `_Char_types<_CharT>::pos_type` **pos_type**
- typedef `_Char_types<_CharT>::state_type` **state_type**

Static Public Member Functions

- static constexpr void **assign** (char_type &__c1, const char_type &__c2)
- static constexpr char_type * **assign** (char_type *__s, std::size_t __n, char_type __a)
- static constexpr int **compare** (const char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr char_type * **copy** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2)
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static constexpr const char_type * **find** (const char_type *__s, std::size_t __n, const char_type &__a)
- static constexpr std::size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2)
- static constexpr char_type * **move** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c)
- static constexpr char_type **to_char_type** (const int_type &__c)
- static constexpr int_type **to_int_type** (const char_type &__c)

6.296.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::char_traits<_CharT>
```

Base class used to implement std::char_traits.

Note

For any given actual character type, this definition is probably wrong. (Most of the member functions are likely to be right, but the int_type and state_type typedefs, and the eof() member function, are likely to be wrong.) The reason this class exists is so users can specialize it. Classes in namespace std may not be specialized for fundamental types, but classes in namespace __gnu_cxx may be.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/strings.html#strings.string.character_types for advice on how to make use of this class for *unusual* character types. Also, check out include/ext/pod_char_traits.h.

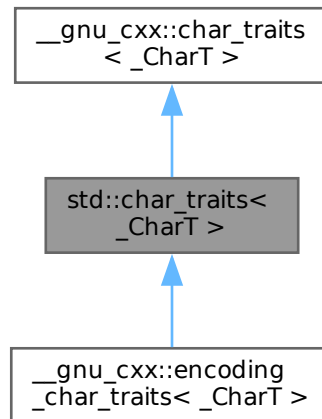
The documentation for this struct was generated from the following file:

- [char_traits.h](#)

6.297 std::char_traits<_CharT> Struct Template Reference

```
#include <char_traits.h>
```

Inheritance diagram for std::char_traits<_CharT>:



Public Types

- typedef _CharT **char_type**
- typedef _Char_types<_CharT>::int_type **int_type**
- typedef _Char_types<_CharT>::off_type **off_type**
- typedef _Char_types<_CharT>::pos_type **pos_type**
- typedef _Char_types<_CharT>::state_type **state_type**

Static Public Member Functions

- static constexpr void **assign** (char_type &__c1, const char_type &__c2)
- static constexpr char_type * **assign** (char_type * __s, std::size_t __n, char_type __a)
- static constexpr int **compare** (const char_type * __s1, const char_type * __s2, std::size_t __n)
- static constexpr char_type * **copy** (char_type * __s1, const char_type * __s2, std::size_t __n)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2)
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static constexpr const char_type * **find** (const char_type * __s, std::size_t __n, const char_type &__a)
- static constexpr std::size_t **length** (const char_type * __s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2)
- static constexpr char_type * **move** (char_type * __s1, const char_type * __s2, std::size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c)
- static constexpr char_type **to_char_type** (const int_type &__c)
- static constexpr int_type **to_int_type** (const char_type &__c)

6.297.1 Detailed Description

```
template<typename _CharT>
struct std::char_traits< _CharT >
```

Basis for explicit traits specializations.

Note

For any given actual character type, this definition is probably wrong. Since this is just a thin wrapper around `__gnu_cxx::char_traits`, it is possible to achieve a more appropriate definition by specializing `__gnu_cxx::char_traits`.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/strings.html#strings.string.character_types for advice on how to make use of this class for *unusual* character types. Also, check out `include/ext/pod_char_traits.h`.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

6.298 std::char_traits< __gnu_cxx::character< _Value, _Int, _St > > Struct Template Reference

```
#include <pod_char_traits.h>
```

Public Types

- typedef `__gnu_cxx::character< _Value, _Int, _St >` **char_type**
- typedef char_type::int_type **int_type**
- typedef `streamoff` **off_type**
- typedef `fpos< state_type >` **pos_type**
- typedef char_type::state_type **state_type**

Static Public Member Functions

- static void **assign** (char_type &__c1, const char_type &__c2)
- static char_type * **assign** (char_type * __s, size_t __n, char_type __a)
- static int **compare** (const char_type * __s1, const char_type * __s2, size_t __n)
- static char_type * **copy** (char_type * __s1, const char_type * __s2, size_t __n)

- static int_type **eof** ()
- static bool **eq** (const char_type &__c1, const char_type &__c2)
- static bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static const char_type * **find** (const char_type * __s, size_t __n, const char_type &__a)
- static size_t **length** (const char_type * __s)
- static bool **lt** (const char_type &__c1, const char_type &__c2)
- static char_type * **move** (char_type * __s1, const char_type * __s2, size_t __n)
- static int_type **not_eof** (const int_type &__c)
- static char_type **to_char_type** (const int_type &__i)
- static int_type **to_int_type** (const char_type &__c)

6.298.1 Detailed Description

```
template<typename _Value, typename _Int, typename _St>
struct std::char_traits< __gnu_cxx::character< _Value, _Int, _St > >
```

char_traits<__gnu_cxx::character> specialization.

The documentation for this struct was generated from the following file:

- [pod_char_traits.h](#)

6.299 std::char_traits< char > Struct Reference

```
#include <char_traits.h>
```

Public Types

- typedef char **char_type**
- typedef int **int_type**
- typedef streamoff **off_type**
- typedef streampos **pos_type**
- typedef mbstate_t **state_type**

Static Public Member Functions

- static constexpr void **assign** (char_type &__c1, const char_type &__c2) noexcept
- static constexpr char_type * **assign** (char_type * __s, size_t __n, char_type __a)
- static constexpr int **compare** (const char_type * __s1, const char_type * __s2, size_t __n)
- static constexpr char_type * **copy** (char_type * __s1, const char_type * __s2, size_t __n)
- static constexpr int_type **eof** () noexcept
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2) noexcept
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2) noexcept
- static constexpr const char_type * **find** (const char_type * __s, size_t __n, const char_type &__a)
- static constexpr size_t **length** (const char_type * __s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2) noexcept
- static constexpr char_type * **move** (char_type * __s1, const char_type * __s2, size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c) noexcept
- static constexpr char_type **to_char_type** (const int_type &__c) noexcept
- static constexpr int_type **to_int_type** (const char_type &__c) noexcept

6.299.1 Detailed Description

21.1.3.1 char_traits specializations

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

6.300 `std::char_traits< wchar_t >` Struct Reference

```
#include <char_traits.h>
```

Public Types

- typedef `wchar_t` **char_type**
- typedef `wint_t` **int_type**
- typedef `streamoff` **off_type**
- typedef `wstreampos` **pos_type**
- typedef `mbstate_t` **state_type**

Static Public Member Functions

- static constexpr void **assign** (`char_type` &__c1, const `char_type` &__c2) noexcept
- static constexpr `char_type` * **assign** (`char_type` * __s, `size_t` __n, `char_type` __a)
- static constexpr int **compare** (const `char_type` * __s1, const `char_type` * __s2, `size_t` __n)
- static constexpr `char_type` * **copy** (`char_type` * __s1, const `char_type` * __s2, `size_t` __n)
- static constexpr `int_type` **eof** () noexcept
- static constexpr bool **eq** (const `char_type` &__c1, const `char_type` &__c2) noexcept
- static constexpr bool **eq_int_type** (const `int_type` &__c1, const `int_type` &__c2) noexcept
- static constexpr const `char_type` * **find** (const `char_type` * __s, `size_t` __n, const `char_type` &__a)
- static constexpr `size_t` **length** (const `char_type` * __s)
- static constexpr bool **lt** (const `char_type` &__c1, const `char_type` &__c2) noexcept
- static constexpr `char_type` * **move** (`char_type` * __s1, const `char_type` * __s2, `size_t` __n)
- static constexpr `int_type` **not_eof** (const `int_type` &__c) noexcept
- static constexpr `char_type` **to_char_type** (const `int_type` &__c) noexcept
- static constexpr `int_type` **to_int_type** (const `char_type` &__c) noexcept

6.300.1 Detailed Description

21.1.3.2 `char_traits` specializations

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

6.301 `__gnu_cxx::character< _Value, _Int, _St >` Struct Template Reference

```
#include <pod_char_traits.h>
```

Public Types

- typedef `character< _Value, _Int, _St >` **char_type**
- typedef `_Int` **int_type**
- typedef `_St` **state_type**
- typedef `_Value` **value_type**

Static Public Member Functions

- template<typename V2 >
static `char_type` **from** (const V2 &v)
- template<typename V2 >
static V2 **to** (const `char_type` &c)

Public Attributes

- `value_type` `value`

6.301.1 Detailed Description

```
template<typename _Value, typename _Int, typename _St = std::mbstate_t>
struct __gnu_cxx::character<_Value, _Int, _St>
```

A POD class that serves as a character abstraction class.

The documentation for this struct was generated from the following file:

- [pod_char_traits.h](#)

6.302 std::chi_squared_distribution<_RealType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- **chi_squared_distribution** (`_RealType` `__n`)
- **chi_squared_distribution** (const [param_type](#) &`__p`)
- template<typename `_ForwardIterator`, typename `_UniformRandomNumberGenerator`>
void **generate** (`_ForwardIterator` `__f`, `_ForwardIterator` `__t`, `_UniformRandomNumberGenerator` &`__urng`)
- template<typename `_ForwardIterator`, typename `_UniformRandomNumberGenerator`>
void **generate** (`_ForwardIterator` `__f`, `_ForwardIterator` `__t`, `_UniformRandomNumberGenerator` &`__urng`, const [param_type](#) &`__p`)
- template<typename `_UniformRandomNumberGenerator`>
void **generate** ([result_type](#) *`__f`, [result_type](#) *`__t`, `_UniformRandomNumberGenerator` &`__urng`)
- template<typename `_UniformRandomNumberGenerator`>
void **generate** ([result_type](#) *`__f`, [result_type](#) *`__t`, `_UniformRandomNumberGenerator` &`__urng`, const [param_type](#) &`__p`)
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- `_RealType` **n** () const
- template<typename `_UniformRandomNumberGenerator`>
[result_type](#) **operator()** (`_UniformRandomNumberGenerator` &`__urng`)
- template<typename `_UniformRandomNumberGenerator`>
[result_type](#) **operator()** (`_UniformRandomNumberGenerator` &`__urng`, const [param_type](#) &`__p`)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &`__param`)
- void **reset** ()

Friends

- `template<typename _RealType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::chi_squared_distribution< _RealType1 > &__x)`
- `bool operator== (const chi_squared_distribution &__d1, const chi_squared_distribution &__d2)`
- `template<typename _RealType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::chi_squared_distribution< _RealType1 > &__x)`

6.302.1 Detailed Description

`template<typename _RealType = double>`
`class std::chi_squared_distribution< _RealType >`

A `chi_squared_distribution` random number distribution.

The formula for the normal probability mass function is $p(x|n) = \frac{x^{(n/2)-1} e^{-x/2}}{\Gamma(n/2) 2^{n/2}}$

6.302.2 Member Typedef Documentation

result_type

`template<typename _RealType = double>`
`typedef _RealType std::chi_squared_distribution< _RealType >::result_type`
 The type of the range of the distribution.

6.302.3 Member Function Documentation

max()

`template<typename _RealType = double>`
`result_type std::chi_squared_distribution< _RealType >::max () const [inline]`
 Returns the least upper bound value of the distribution.
 References `std::numeric_limits< _Tp >::max()`.

min()

`template<typename _RealType = double>`
`result_type std::chi_squared_distribution< _RealType >::min () const [inline]`
 Returns the greatest lower bound value of the distribution.

operator()()

`template<typename _RealType = double>`
`template<typename _UniformRandomNumberGenerator >`
`result_type std::chi_squared_distribution< _RealType >::operator() (`
`_UniformRandomNumberGenerator & __urng) [inline]`
 Generating functions.

param() [1/2]

`template<typename _RealType = double>`
`param_type std::chi_squared_distribution< _RealType >::param () const [inline]`
 Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::chi_squared_distribution<_RealType>::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

References [std::gamma_distribution<_RealType>::param\(\)](#).

reset()

```
template<typename _RealType = double>
void std::chi_squared_distribution<_RealType>::reset ( ) [inline]
```

Resets the distribution state.

References [std::gamma_distribution<_RealType>::reset\(\)](#).

6.302.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream<_CharT, _Traits> & operator<< (
    std::basic_ostream<_CharT, _Traits> & __os,
    const std::chi_squared_distribution<_RealType1> & __x ) [friend]
```

Inserts a `chi_squared_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>chi_squared_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const chi_squared_distribution<_RealType> & __d1,
    const chi_squared_distribution<_RealType> & __d2 ) [friend]
```

Return true if two Chi-squared distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream<_CharT, _Traits> & operator>> (
```

```
std::basic_istream< _CharT, _Traits > & __is,
std::chi_squared_distribution< _RealType1 > & __x ) [friend]
```

Extracts a `chi_squared_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>chi_squared_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

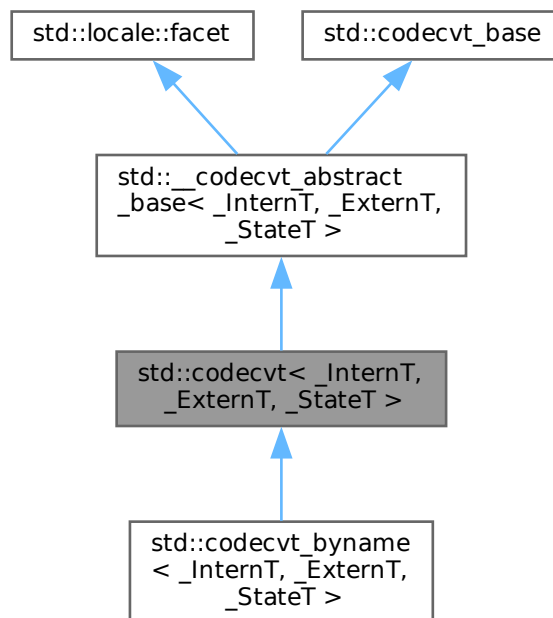
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.303 `std::codecvt< _InternT, _ExternT, _StateT >` Class Template Reference

```
#include <codecvt.h>
```

Inheritance diagram for `std::codecvt< _InternT, _ExternT, _StateT >`:



Public Types

- typedef `_ExternT` **extern_type**

- typedef _InternT **intern_type**
- typedef codecvt_base::result **result**
- typedef _StateT **state_type**

Public Member Functions

- **codecvt** (__c_locale __cloc, size_t __refs=0)
- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_codecvt**

6.303.1 Detailed Description

template<typename _InternT, typename _ExternT, typename _StateT>
class std::codecvt<_InternT, _ExternT, _StateT >

Primary class template codecvt.

NB: Generic, mostly useless implementation.

6.303.2 Member Function Documentation

do_always_noconv()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual bool std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_always\_noconv \( \) const throw ( )
[protected], [virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

do_encoding()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_encoding \( \) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

do_in()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_in \(
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

do_length()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_length \(
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

do_max_length()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_max\_length \( \) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

do_out()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_out \(
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
```

```
extern_type * __to_end,
extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

do_unshift()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::__codecvt<_InternT, _ExternT, _StateT >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

in()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The state argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how state is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

out()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type * __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

References [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::do_out\(\)](#).

unshift()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

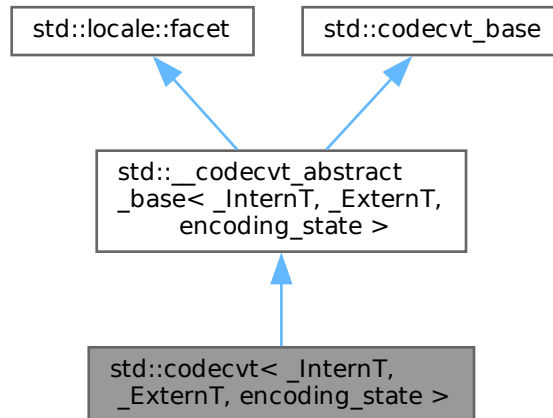
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.304 std::codecvt< _InternT, _ExternT, encoding_state > Class Template Reference

```
#include <codecvt_specializations.h>
```

Inheritance diagram for `std::codecvt< _InternT, _ExternT, encoding_state >`:



Public Types

- `typedef state_type::descriptor_type` **descriptor_type**

- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `__gnu_cxx::encoding_state` **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- **codecvt** (state_type &__enc, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static `locale::id` **id**

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to↔_next) const

Static Protected Member Functions

- static `__c_locale` **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static `__c_locale` **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static `__c_locale` **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

6.304.1 Detailed Description

`template<typename _InternT, typename _ExternT>`
`class std::codecvt< _InternT, _ExternT, encoding_state >`

`codecvt<InternT, _ExternT, encoding_state>` specialization.

6.304.2 Member Function Documentation

do_always_noconv()

```
template<typename _InternT , typename _ExternT >
bool std::codecvt< _InternT, _ExternT, encoding_state >::do_always_noconv throw ( )    [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >](#).

do_encoding()

```
template<typename _InternT , typename _ExternT >
int std::codecvt< _InternT, _ExternT, encoding_state >::do_encoding throw ( )    [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >](#).

do_in()

```
template<typename _InternT , typename _ExternT >
codecvt_base::result std::codecvt< _InternT, _ExternT, encoding_state >::do_in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >](#).

do_length()

```
template<typename _InternT , typename _ExternT >
int std::codecvt< _InternT, _ExternT, encoding_state >::do_length (
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >](#).

do_max_length()

```
template<typename _InternT , typename _ExternT >
int std::codecvt< _InternT, _ExternT, encoding_state >::do_max_length throw ( )    [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >](#).

do_out()

```
template<typename _InternT , typename _ExternT >
codecvt_base::result std::codecvt< _InternT, _ExternT, encoding_state >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
```

```
extern_type * __to_end,
extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements `std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >`.

do_unshift()

```
template<typename _InternT , typename _ExternT >
codecvt_base::result std::__codecvt< _InternT, _ExternT, encoding_state >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Implements `std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >`.

in()

```
result std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

out()

```
result std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type * __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

unshift()

```
result std::__codecvt_abstract_base< _InternT, _ExternT, encoding_state >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type * __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling codecvt::do_unshift().

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions. The source and destination character sets are determined by the facet's locale, internal and external types. The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

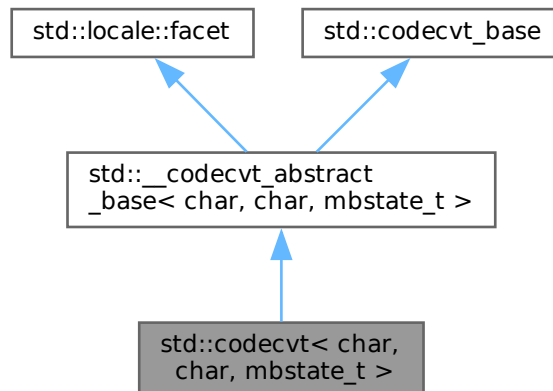
The documentation for this class was generated from the following file:

- [codecvt_specializations.h](#)

6.305 `std::codecvt< char, char, mbstate_t >` Class Reference

```
#include <codecvt.h>
```

Inheritance diagram for `std::codecvt< char, char, mbstate_t >`:



Public Types

- typedef char **extern_type**
- typedef char **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `mbstate_t` **state_type**

Public Member Functions

- **codecvt** (__c_locale __cloc, size_t __refs=0)
- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to↵_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_type_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_codecvt**

Friends

- class **messages< char >**

6.305.1 Detailed Description

class codecvt<char, char, mbstate_t> specialization.

6.305.2 Member Function Documentation

do_always_noconv()

```
virtual bool std::codecvt< char, char, mbstate_t >::do_always_noconv ( ) const throw ( )    [protected],  
[virtual]
```

Implements [std::__codecvt_abstract_base](#)< char, char, mbstate_t >.

do_encoding()

```
virtual int std::codecvt< char, char, mbstate_t >::do_encoding ( ) const throw ( )    [protected],  
[virtual]
```

Implements [std::__codecvt_abstract_base](#)< char, char, mbstate_t >.

do_in()

```
virtual result std::codecvt< char, char, mbstate_t >::do_in (  
    state_type & __state,  
    const extern_type * __from,  
    const extern_type * __from_end,  
    const extern_type *& __from_next,  
    intern_type * __to,  
    intern_type * __to_end,  
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base](#)< char, char, mbstate_t >.

do_length()

```
virtual int std::codecvt< char, char, mbstate_t >::do_length (  
    state_type & ,  
    const extern_type * __from,  
    const extern_type * __end,  
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base](#)< char, char, mbstate_t >.

do_max_length()

```
virtual int std::codecvt< char, char, mbstate_t >::do_max_length ( ) const throw ( )    [protected],  
[virtual]
```

Implements [std::__codecvt_abstract_base](#)< char, char, mbstate_t >.

do_out()

```
virtual result std::codecvt< char, char, mbstate_t >::do_out (  
    state_type & __state,  
    const intern_type * __from,  
    const intern_type * __from_end,  
    const intern_type *& __from_next,  
    extern_type * __to,  
    extern_type * __to_end,  
    extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base< char, char, mbstate_t >](#).

do_unshift()

```
virtual result std::codecvt< char, char, mbstate_t >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char, char, mbstate_t >](#).

in()

```
result std::__codecvt_abstract_base< char , char , mbstate_t >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

out()

```
result std::__codecvt_abstract_base< char , char , mbstate_t >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

unshift()

```
result std::__codecvt_abstract_base< char , char , mbstate_t >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling codecvt::do_unshift().

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions. The source and destination character sets are determined by the facet's locale, internal and external types. The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

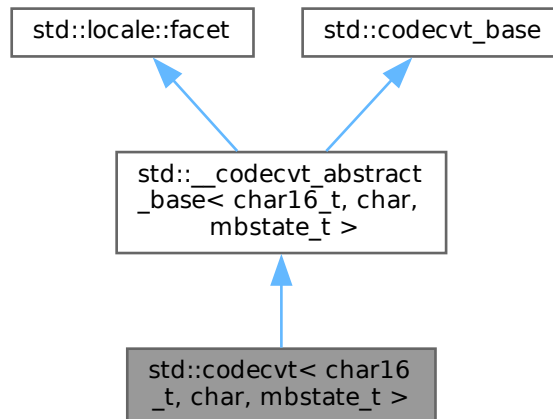
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.306 std::codecvt< char16_t, char, mbstate_t > Class Reference

```
#include <codecvt.h>
```

Inheritance diagram for `std::codecvt< char16_t, char, mbstate_t >`:



Public Types

- typedef char **extern_type**
- typedef char16_t **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `mbstate_t` **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual bool [do_always_noconv](#) () const throw ()
- virtual int [do_encoding](#) () const throw ()
- virtual result [do_in](#) (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int [do_length](#) (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int [do_max_length](#) () const throw ()
- virtual result [do_out](#) (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result [do_unshift](#) (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to↵__next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

6.306.1 Detailed Description

Class `codecvt<char16_t, char, mbstate_t>` specialization.
Converts between UTF-16 and UTF-8.

6.306.2 Member Function Documentation

do_always_noconv()

```
virtual bool std::codecvt< char16_t, char, mbstate_t >::do_always_noconv ( ) const throw ( )  
[protected], [virtual]
```

Implements [std::__codecvt_abstract_base](#)< char16_t, char, mbstate_t >.

do_encoding()

```
virtual int std::codecvt< char16_t, char, mbstate_t >::do_encoding ( ) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

do_in()

```
virtual result std::codecvt< char16_t, char, mbstate_t >::do_in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

do_length()

```
virtual int std::codecvt< char16_t, char, mbstate_t >::do_length (
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

do_max_length()

```
virtual int std::codecvt< char16_t, char, mbstate_t >::do_max_length ( ) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

do_out()

```
virtual result std::codecvt< char16_t, char, mbstate_t >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

[do_out](#) for more information.

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

do_unshift()

```
virtual result std::codecvt< char16_t, char, mbstate_t >::do_unshift (
    state_type & __state,
```



```

extern_type * __to,
extern_type * __to_end,
extern_type *& __to_next ) const [protected], [virtual]

```

Implements `std::__codecvt_abstract_base< char16_t, char, mbstate_t >`.

in()

```

result std::__codecvt_abstract_base< char16_t , char , mbstate_t >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]

```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

out()

```

result std::__codecvt_abstract_base< char16_t , char , mbstate_t >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]

```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

unshift()

```
result std::__codecvt_abstract_base< char16_t , char , mbstate_t >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore `state` to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

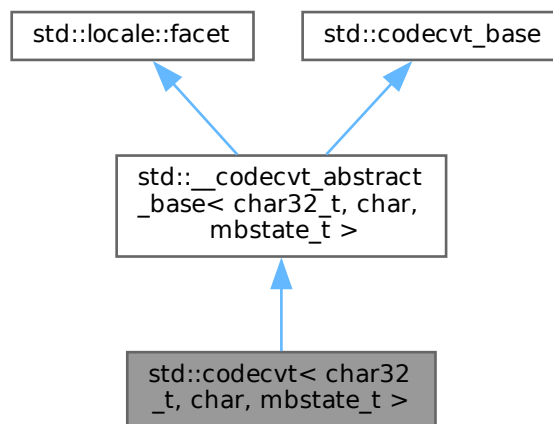
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.307 std::codecvt< char32_t, char, mbstate_t > Class Reference

```
#include <codecvt.h>
```

Inheritance diagram for std::codecvt< char32_t, char, mbstate_t >:

**Public Types**

- typedef char **extern_type**
- typedef char32_t **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual bool [do_always_noconv](#) () const throw ()
- virtual int [do_encoding](#) () const throw ()
- virtual result [do_in](#) (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *&__from_next, intern_type *__to, intern_type *__to_end, intern_type *&__to_next) const
- virtual int [do_length](#) (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int [do_max_length](#) () const throw ()
- virtual result [do_out](#) (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *&__from_next, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const
- virtual result [do_unshift](#) (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale &__cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale &__cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_type_c_locale](#) (__c_locale __cloc, const char *__s)

6.307.1 Detailed Description

Class `codecvt<char32_t, char, mbstate_t>` specialization.
Converts between UTF-32 and UTF-8.

6.307.2 Member Function Documentation

`do_always_noconv()`

```
virtual bool std::codecvt< char32_t, char, mbstate_t >::do_always_noconv ( ) const throw ( )
[protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

`do_encoding()`

```
virtual int std::codecvt< char32_t, char, mbstate_t >::do_encoding ( ) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

`do_in()`

```
virtual result std::codecvt< char32_t, char, mbstate_t >::do_in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

do_length()

```
virtual int std::codecvt< char32_t, char, mbstate_t >::do_length (
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

do_max_length()

```
virtual int std::codecvt< char32_t, char, mbstate_t >::do_max_length ( ) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

do_out()

```
virtual result std::codecvt< char32_t, char, mbstate_t >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

[do_out](#) for more information.

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

do_unshift()

```
virtual result std::codecvt< char32_t, char, mbstate_t >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< char32_t, char, mbstate_t >](#).

in()

```
result std::__codecvt_abstract_base< char32_t , char , mbstate_t >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

out()

```
result std::__codecvt_abstract_base< char32_t , char , mbstate_t >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.

Parameters

<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

unshift()

```
result std::__codecvt_abstract_base< char32_t , char , mbstate_t >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

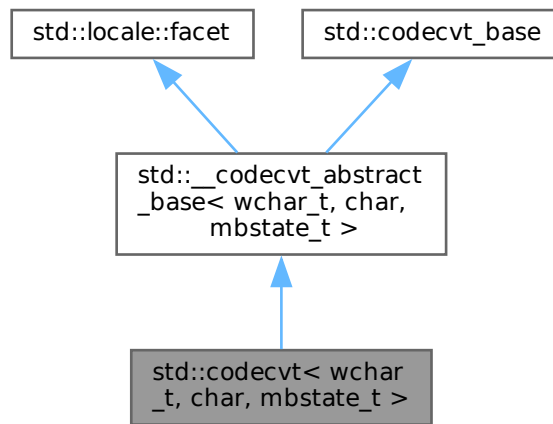
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.308 std::codecvt< wchar_t, char, mbstate_t > Class Reference

```
#include <codecvt.h>
```

Inheritance diagram for std::codecvt< wchar_t, char, mbstate_t >:



Public Types

- typedef char **extern_type**
- typedef wchar_t **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (`__c_locale __cloc, size_t __refs=0`)
- **codecvt** (`size_t __refs=0`)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (`state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__from_next, intern_type *__to, intern_type *__to_end, intern_type *__to_next`) const
- int **length** (`state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max`) const
- int **max_length** () const throw ()
- result **out** (`state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__from_next, extern_type *__to, extern_type *__to_end, extern_type *__to_next`) const
- result **unshift** (`state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__to_next`) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `bool do_always_noconv()` `const throw()`
- virtual `int do_encoding()` `const throw()`
- virtual `result do_in(state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type * &__from_next, intern_type *__to, intern_type *__to_end, intern_type * &__to_next)` `const`
- virtual `int do_length(state_type &, const extern_type *__from, const extern_type *__end, size_t __max)` `const`
- virtual `int do_max_length()` `const throw()`
- virtual `result do_out(state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type * &__from_next, extern_type *__to, extern_type *__to_end, extern_type * &__to_next)` `const`
- virtual `result do_unshift(state_type &__state, extern_type *__to, extern_type *__to_end, extern_type * &__to←__next)` `const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale &__cloc)` `throw()`
- static `void _S_create_c_locale(__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static `void _S_destroy_c_locale(__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale()`
- static `const char * _S_get_c_name()` `throw()`
- static `__c_locale _S_lc_ctype_c_locale(__c_locale __cloc, const char *__s)`

Protected Attributes

- `__c_locale _M_c_locale_codecvt`

Friends

- class `messages<wchar_t>`

6.308.1 Detailed Description

Class `codecvt<wchar_t, char, mbstate_t>` specialization.
Converts between narrow and wide characters in the native character set

6.308.2 Member Function Documentation

`do_always_noconv()`

```
virtual bool std::codecvt< wchar_t, char, mbstate_t >::do_always_noconv ( ) const throw ( )    [protected],
[virtual]
```

Implements `std::__codecvt_abstract_base< wchar_t, char, mbstate_t >`.

`do_encoding()`

```
virtual int std::codecvt< wchar_t, char, mbstate_t >::do_encoding ( ) const throw ( )    [protected],
[virtual]
```

Implements `std::__codecvt_abstract_base< wchar_t, char, mbstate_t >`.

do_in()

```
virtual result std::codecvt< wchar_t, char, mbstate_t >::do_in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

do_length()

```
virtual int std::codecvt< wchar_t, char, mbstate_t >::do_length (
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

do_max_length()

```
virtual int std::codecvt< wchar_t, char, mbstate_t >::do_max_length ( ) const throw ( ) [protected],
[virtual]
```

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

do_out()

```
virtual result std::codecvt< wchar_t, char, mbstate_t >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

[do_out\(\)](#) for more information.

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

do_unshift()

```
virtual result std::codecvt< wchar_t, char, mbstate_t >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual]
```

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

in()

```
result std::__codecvt_abstract_base< wchar_t , char , mbstate_t >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

out()

```
result std::__codecvt_abstract_base< wchar_t , char , mbstate_t >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

unshift()

```
result std::__codecvt_abstract_base< wchar_t , char , mbstate_t >::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

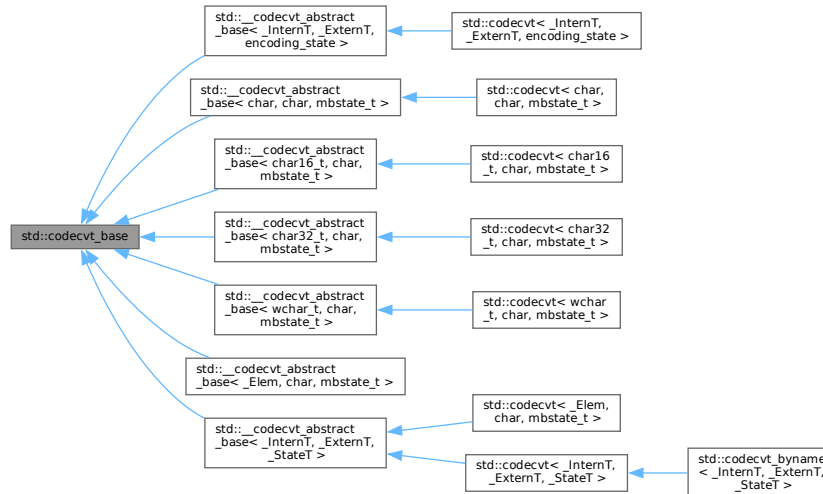
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.309 std::codecvt_base Class Reference

```
#include <codecvt.h>
```

Inheritance diagram for std::codecvt_base:



Public Types

- enum **result** { **ok** , **partial** , **error** , **noconv** }

6.309.1 Detailed Description

Empty base class for codecvt facet [22.2.1.5].

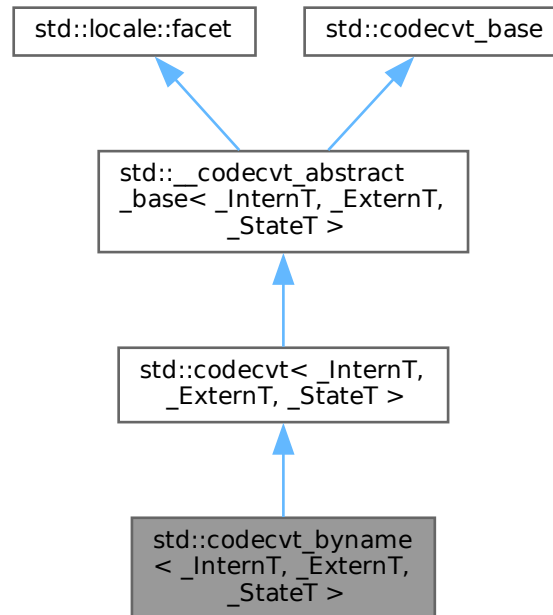
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.310 std::codecvt_byname<_InternT, _ExternT, _StateT> Class Template Reference

```
#include <codecvt.h>
```

Inheritance diagram for std::codecvt_byname< _InternT, _ExternT, _StateT >:



Public Types

- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `_StateT` **state_type**

Public Member Functions

- **codecvt_byname** (const char *__s, size_t __refs=0)
- **codecvt_byname** (const [string](#) &__s, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) **id**

Protected Member Functions

- virtual bool [do_always_noconv](#) () const throw ()
- virtual int [do_encoding](#) () const throw ()
- virtual result [do_in](#) (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int [do_length](#) (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int [do_max_length](#) () const throw ()
- virtual result [do_out](#) (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result [do_unshift](#) (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale &__cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale &__cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale [_M_c_locale_codecvt](#)

6.310.1 Detailed Description

```
template<typename _InternT, typename _ExternT, typename _StateT>
class std::codecvt_byname< _InternT, _ExternT, _StateT >
```

class codecvt_byname [22.2.1.6].

6.310.2 Member Function Documentation

do_always_noconv()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual bool std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_always\_noconv ( ) const throw ( )
[protected], [virtual], [inherited]
Implements std::\_\_codecvt\_abstract\_base< \_InternT, \_ExternT, \_StateT >.
```

do_encoding()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_encoding ( ) const throw ( ) [protected],
[virtual], [inherited]
Implements std::\_\_codecvt\_abstract\_base< \_InternT, \_ExternT, \_StateT >.
```

do_in()

```
template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::codecvt< \_InternT, \_ExternT, \_StateT >::do\_in (
    state_type & __state,
    const extern_type * __from,
```

```

    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [protected], [virtual], [inherited]

```

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

do_length()

```

template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt<_InternT, _ExternT, _StateT >::do_length (
    state_type & ,
    const extern_type * __from,
    const extern_type * __end,
    size_t __max ) const [protected], [virtual], [inherited]

```

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

do_max_length()

```

template<typename _InternT , typename _ExternT , typename _StateT >
virtual int std::codecvt<_InternT, _ExternT, _StateT >::do_max_length ( ) const throw ( ) [protected],
[virtual], [inherited]

```

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

do_out()

```

template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::codecvt<_InternT, _ExternT, _StateT >::do_out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual], [inherited]

```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

[do_out](#) for more information.

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

do_unshift()

```

template<typename _InternT , typename _ExternT , typename _StateT >
virtual result std::codecvt<_InternT, _ExternT, _StateT >::do_unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [protected], [virtual], [inherited]

```

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

in()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::in (
    state_type & __state,
    const extern_type * __from,
    const extern_type * __from_end,
    const extern_type *& __from_next,
    intern_type * __to,
    intern_type * __to_end,
    intern_type *& __to_next ) const [inline], [inherited]
```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

out()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::out (
    state_type & __state,
    const intern_type * __from,
    const intern_type * __from_end,
    const intern_type *& __from_next,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

References [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::do_out\(\)](#).

unshift()

```
template<typename _InternT , typename _ExternT , typename _StateT >
result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::unshift (
    state_type & __state,
    extern_type * __to,
    extern_type * __to_end,
    extern_type *& __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

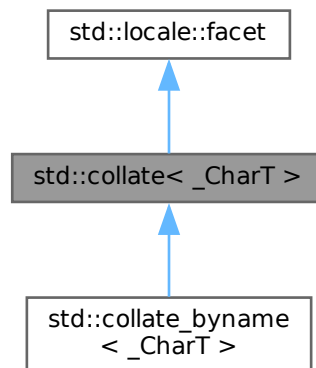
The documentation for this class was generated from the following file:

- [codecvt.h](#)

6.311 std::collate<_CharT> Class Template Reference

```
#include <locale_classes.h>
```

Inheritance diagram for std::collate<_CharT>:



Public Types

- typedef `_CharT` [char_type](#)
- typedef [basic_string](#)<_CharT> [string_type](#)

Public Member Functions

- [collate](#) (`__c_locale __cloc, size_t __refs=0`)
- [collate](#) (`size_t __refs=0`)
- `int` [_M_compare](#) (`const _CharT *, const _CharT *`) `const throw ()`
- `int` [_M_compare](#) (`const char *, const char *`) `const throw()`
- `int` [_M_compare](#) (`const wchar_t *, const wchar_t *`) `const throw()`
- `size_t` [_M_transform](#) (`_CharT *, const _CharT *, size_t`) `const throw ()`
- `size_t` [_M_transform](#) (`char *, const char *, size_t`) `const throw()`
- `size_t` [_M_transform](#) (`wchar_t *, const wchar_t *, size_t`) `const throw()`
- `int` [compare](#) (`const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2`) `const`
- `long` [hash](#) (`const _CharT * __lo, const _CharT * __hi`) `const`
- [string_type transform](#) (`const _CharT * __lo, const _CharT * __hi`) `const`

Static Public Attributes

- static [locale::id](#) `id`

Protected Member Functions

- virtual [~collate](#) ()
- virtual int [do_compare](#) (const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2) const
- virtual long [do_hash](#) (const _CharT * __lo, const _CharT * __hi) const
- virtual [string_type do_transform](#) (const _CharT * __lo, const _CharT * __hi) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_type_c_locale](#) (__c_locale __cloc, const char * __s)

Protected Attributes

- __c_locale [_M_c_locale_collate](#)

6.311.1 Detailed Description

template<typename _CharT>
class std::collate<_CharT>

Facet for localized string comparison.

This facet encapsulates the code to compare strings in a localized manner.

The collate template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the collate facet.

6.311.2 Member Typedef Documentation**char_type**

```
template<typename _CharT >
typedef _CharT std::collate<\_CharT>::char\_type
Public typedefs.
```

string_type

```
template<typename _CharT >
typedef basic\_string<\_CharT> std::collate<\_CharT>::string\_type
Public typedefs.
```

6.311.3 Constructor & Destructor Documentation**collate()** [1/2]

```
template<typename _CharT >
std::collate<\_CharT>::collate (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

collate() [2/2]

```
template<typename _CharT >
std::collate< _CharT >::collate (
    __c_locale __cloc,
    size_t __refs = 0 ) [inline], [explicit]
```

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__refs</code>	Passed to the base facet class.

~collate()

```
template<typename _CharT >
virtual std::collate< _CharT >::~~collate ( ) [inline], [protected], [virtual]
```

Destructor.

6.311.4 Member Function Documentation**compare()**

```
template<typename _CharT >
int std::collate< _CharT >::compare (
    const _CharT * __lo1,
    const _CharT * __hi1,
    const _CharT * __lo2,
    const _CharT * __hi2 ) const [inline]
```

Compare two strings.

This function compares two strings and returns the result by calling `collate::do_compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

do_compare()

```
template<typename _CharT >
int std::collate< _CharT >::do_compare (
```

```

    const _CharT * __lo1,
    const _CharT * __hi1,
    const _CharT * __lo2,
    const _CharT * __hi2 ) const [protected], [virtual]

```

Compare two strings.

This function is a hook for derived classes to change the value returned.

See also

`compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

References [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::length\(\)](#).

`do_hash()`

```

template<typename _CharT >
long std::collate<_CharT>::do_hash (
    const _CharT * __lo,
    const _CharT * __hi ) const [protected], [virtual]

```

Return hash of a string.

This function computes and returns a hash on the input string. This function is a hook for derived classes to change the value returned.

Parameters

<code>__lo</code>	Start of string.
<code>__hi</code>	End of string.

Returns

Hash value.

`do_transform()`

```

template<typename _CharT >
collate<_CharT>::string_type std::collate<_CharT>::do_transform (
    const _CharT * __lo,
    const _CharT * __hi ) const [protected], [virtual]

```

Transform string to comparable form.

This function is a hook for derived classes to change the value returned.

Parameters

<code>_↔ _lo</code>	Start.
<code>_↔ _hi</code>	End.

Returns

transformed string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::length\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::push_back\(\)](#).

hash()

```
template<typename _CharT >
long std::collate<_CharT>::hash (
    const _CharT * __lo,
    const _CharT * __hi ) const [inline]
```

Return hash of a string.

This function computes and returns a hash on the input string. It does so by returning `collate::do_hash()`.

Parameters

<code>_↔ _lo</code>	Start of string.
<code>_↔ _hi</code>	End of string.

Returns

Hash value.

transform()

```
template<typename _CharT >
string_type std::collate<_CharT>::transform (
    const _CharT * __lo,
    const _CharT * __hi ) const [inline]
```

Transform string to comparable form.

This function is a wrapper for `strxfrm` functionality. It takes the input string and returns a modified string that can be directly compared to other transformed strings. In the C locale, this function just returns a copy of the input string. In some other locales, it may replace two chars with one, change a char for another, etc. It does so by returning `collate::do_transform()`.

Parameters

<code>_↔ _lo</code>	Start of string.
<code>_↔ _hi</code>	End of string.

Returns

Transformed `string_type`.

6.311.5 Member Data Documentation

id

```
template<typename _CharT >
locale::id std::collate< _CharT >::id [static]
Numpunct facet id.
```

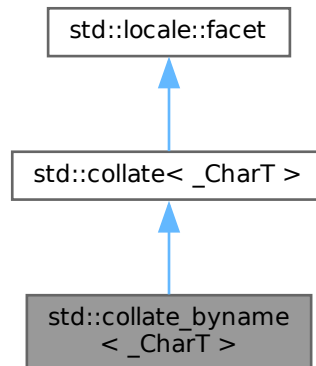
The documentation for this class was generated from the following files:

- [locale_classes.h](#)
- [locale_classes.tcc](#)

6.312 std::collate_byname< _CharT > Class Template Reference

```
#include <locale_classes.h>
```

Inheritance diagram for `std::collate_byname< _CharT >`:



Public Types

- typedef `_CharT` [char_type](#)
- typedef [basic_string](#)< `_CharT` > [string_type](#)

Public Member Functions

- **collate_byname** (const char *__s, size_t __refs=0)
- **collate_byname** (const [string](#) &__s, size_t __refs=0)
- **_M_compare** (const `_CharT` *, const `_CharT` *) const throw ()
- **_M_compare** (const char *, const char *) const throw()
- **_M_compare** (const `wchar_t` *, const `wchar_t` *) const throw()
- **_M_transform** (`_CharT` *, const `_CharT` *, size_t) const throw ()
- **_M_transform** (char *, const char *, size_t) const throw()

- `size_t M_transform` (`wchar_t *`, `const wchar_t *`, `size_t`) `const throw()`
- `int compare` (`const _CharT *__lo1`, `const _CharT *__hi1`, `const _CharT *__lo2`, `const _CharT *__hi2`) `const`
- `long hash` (`const _CharT *__lo`, `const _CharT *__hi`) `const`
- `string_type transform` (`const _CharT *__lo`, `const _CharT *__hi`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `int do_compare` (`const _CharT *__lo1`, `const _CharT *__hi1`, `const _CharT *__lo2`, `const _CharT *__hi2`) `const`
- virtual `long do_hash` (`const _CharT *__lo`, `const _CharT *__hi`) `const`
- virtual `string_type do_transform` (`const _CharT *__lo`, `const _CharT *__hi`) `const`

Static Protected Member Functions

- static `__c_locale S_clone_c_locale` (`__c_locale &__cloc`) `throw ()`
- static `void S_create_c_locale` (`__c_locale &__cloc`, `const char *__s`, `__c_locale __old=0`)
- static `void S_destroy_c_locale` (`__c_locale &__cloc`)
- static `__c_locale S_get_c_locale` ()
- static `const char * S_get_c_name` () `throw ()`
- static `__c_locale S_lc_ctype_c_locale` (`__c_locale __cloc`, `const char *__s`)

Protected Attributes

- `__c_locale M_c_locale_collate`

6.312.1 Detailed Description

`template<typename _CharT>`
`class std::collate_byname<_CharT>`

class `collate_byname` [22.2.4.2].

6.312.2 Member Typedef Documentation

`char_type`

```
template<typename _CharT >
typedef _CharT std::collate_byname<_CharT>::char_type
```

Public typedefs.

`string_type`

```
template<typename _CharT >
typedef basic_string<_CharT> std::collate_byname<_CharT>::string_type
```

Public typedefs.

6.312.3 Member Function Documentation

compare()

```
template<typename _CharT >
int std::collate< _CharT >::compare (
    const _CharT * __lo1,
    const _CharT * __hi1,
    const _CharT * __lo2,
    const _CharT * __hi2 ) const [inline], [inherited]
```

Compare two strings.

This function compares two strings and returns the result by calling `collate::do_compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

do_compare()

```
template<typename _CharT >
int std::collate< _CharT >::do_compare (
    const _CharT * __lo1,
    const _CharT * __hi1,
    const _CharT * __lo2,
    const _CharT * __hi2 ) const [protected], [virtual], [inherited]
```

Compare two strings.

This function is a hook for derived classes to change the value returned.

See also

`compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

References `std::basic_string< _CharT, _Traits, _Alloc >::c_str()`, `std::basic_string< _CharT, _Traits, _Alloc >::data()`, and `std::basic_string< _CharT, _Traits, _Alloc >::length()`.



do_hash()

```
template<typename _CharT >
long std::collate<_CharT>::do_hash (
    const _CharT * __lo,
    const _CharT * __hi ) const [protected], [virtual], [inherited]
```

Return hash of a string.

This function computes and returns a hash on the input string. This function is a hook for derived classes to change the value returned.

Parameters

_lo	Start of string.
_hi	End of string.

Returns

Hash value.



do_transform()

```
template<typename _CharT >
collate<_CharT>::string_type std::collate<_CharT>::do_transform (
    const _CharT * __lo,
    const _CharT * __hi ) const [protected], [virtual], [inherited]
```

Transform string to comparable form.

This function is a hook for derived classes to change the value returned.

Parameters

_lo	Start.
_hi	End.

Returns

transformed string.

References [std::basic_string<_CharT, _Traits, _Alloc>::append\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::length\(\)](#), and [std::basic_string<_CharT, _Traits, _Alloc>::push_back\(\)](#).

hash()

```
template<typename _CharT >
long std::collate<_CharT>::hash (
    const _CharT * __lo,
    const _CharT * __hi ) const [inline], [inherited]
```

Return hash of a string.

This function computes and returns a hash on the input string. It does so by returning `collate::do_hash()`.

Parameters

<code>_↔ _lo</code>	Start of string.
<code>_↔ _hi</code>	End of string.

Returns

Hash value.

transform()

```
template<typename _CharT >
string_type std::collate< _CharT >::transform (
    const _CharT * __lo,
    const _CharT * __hi ) const [inline], [inherited]
```

Transform string to comparable form.

This function is a wrapper for `strxfrm` functionality. It takes the input string and returns a modified string that can be directly compared to other transformed strings. In the C locale, this function just returns a copy of the input string. In some other locales, it may replace two chars with one, change a char for another, etc. It does so by returning `collate::do_transform()`.

Parameters

<code>_↔ _lo</code>	Start of string.
<code>_↔ _hi</code>	End of string.

Returns

Transformed `string_type`.

6.312.4 Member Data Documentation**id**

```
template<typename _CharT >
locale::id std::collate< _CharT >::id [static], [inherited]
```

Numpunct facet id.

The documentation for this class was generated from the following file:

- [locale_classes.h](#)

6.313 std::common_iterator< _It, _Sent > Class Template Reference

```
#include <stl_iterator.h>
```

Public Member Functions

- constexpr **common_iterator** (`_It __i`) noexcept(is_nothrow_move_constructible_v< `_It` >)
- constexpr **common_iterator** (`_Sent __s`) noexcept(is_nothrow_move_constructible_v< `_Sent` >)
- constexpr **common_iterator** ([common_iterator](#) &&__x) noexcept(`_S_noexcept< _It, _Sent >()`)

- constexpr **common_iterator** (const [common_iterator](#) &__x) noexcept(_S_noexcept< const _It &, const _Sent &>())
- template<typename _It2, typename _Sent2 >
requires convertible_to<const _It2&, _It> && convertible_to<const _Sent2&, _Sent>
constexpr **common_iterator** (const [common_iterator](#)< _It2, _Sent2 > &__x) noexcept(_S_noexcept< const _It2 &, const _Sent2 &>())
- constexpr decltype(auto) **operator*** ()
- constexpr decltype(auto) **operator*** () const
- constexpr [common_iterator](#) & **operator++** ()
- constexpr decltype(auto) **operator++** (int)
- constexpr auto **operator-->** () const
- constexpr [common_iterator](#) & **operator=** ([common_iterator](#) &&)=default
- constexpr [common_iterator](#) & **operator=** ([common_iterator](#) && __x) noexcept(is_nothrow_move_assignable_v< _It > &&is_nothrow_move_assignable_v< _Sent > &&is_nothrow_move_constructible_v< _It > &&is_nothrow_move_constructible_v< _Sent >)
- constexpr [common_iterator](#) & **operator=** (const [common_iterator](#) &)=default
- constexpr [common_iterator](#) & **operator=** (const [common_iterator](#) & __x) noexcept(is_nothrow_copy_assignable_v< _It > &&is_nothrow_copy_assignable_v< _Sent > &&is_nothrow_copy_constructible_v< _It > &&is_nothrow_copy_constructible_v< _Sent >)
- template<typename _It2, typename _Sent2 >
requires convertible_to<const _It2&, _It> && convertible_to<const _Sent2&, _Sent> && assignable_from<_It&, const _It2&> && assignable_from<_Sent&, const _Sent2&>
constexpr [common_iterator](#) & **operator=** (const [common_iterator](#)< _It2, _Sent2 > &__x) noexcept(is_nothrow_constructible_v< _It, const _It2 > &&is_nothrow_constructible_v< _Sent, const _Sent2 > &&is_nothrow_assignable_v< _It &, const _It2 > &&is_nothrow_assignable_v< _Sent &, const _Sent2 > &>)

Friends

- constexpr iter_rvalue_reference_t< _It > **iter_move** (const [common_iterator](#) &__i) noexcept(noexcept(ranges::iter_move(std::declval< const _It &>())))
- template<indirectly_swappable< _It > _It2, typename _Sent2 >
constexpr void **iter_swap** (const [common_iterator](#) &__x, const [common_iterator](#)< _It2, _Sent2 > &__y) noexcept(noexcept(ranges::iter_swap(std::declval< const _It &>(), std::declval< const _It2 &>())))
- template<sized_sentinel_for< _It > _It2, sized_sentinel_for< _It > _Sent2>
requires sized_sentinel_for< _Sent, _It2>
constexpr iter_difference_t< _It2 > **operator-** (const [common_iterator](#) &__x, const [common_iterator](#)< _It2, _Sent2 > &__y)
- template<typename _It2, sentinel_for< _It > _Sent2>
requires sentinel_for< _Sent, _It2>
constexpr bool **operator==** (const [common_iterator](#) &__x, const [common_iterator](#)< _It2, _Sent2 > &__y)
- template<typename _It2, sentinel_for< _It > _Sent2>
requires sentinel_for< _Sent, _It2> && equality_comparable_with< _It, _It2>
constexpr bool **operator==** (const [common_iterator](#) &__x, const [common_iterator](#)< _It2, _Sent2 > &__y)

6.313.1 Detailed Description

template<input_or_output_iterator _It, sentinel_for< _It > _Sent>
requires (!same_as<_It, _Sent>) && copyable<_It>
class std::common_iterator< _It, _Sent >

An iterator/sentinel adaptor for representing a non-common range.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.314 `std::common_type<_Tp>` Struct Template Reference

6.314.1 Detailed Description

```
template<typename... _Tp>
struct std::common_type<_Tp>
```

`common_type`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.315 `std::common_type< chrono::duration<_Rep, _Period>>` Struct Template Reference

```
#include <chrono.h>
```

Public Types

- using **type** = [chrono::duration](#)< typename [common_type](#)<_Rep>::type, typename _Period::type >

6.315.1 Detailed Description

```
template<typename _Rep, typename _Period>
struct std::common_type< chrono::duration<_Rep, _Period>>
```

Specialization of `common_type` for one `chrono::duration` type.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.316 `std::common_type< chrono::duration<_Rep, _Period>, chrono::duration<_Rep, _Period>>` Struct Template Reference

```
#include <chrono.h>
```

Public Types

- using **type** = [chrono::duration](#)< typename [common_type](#)<_Rep>::type, typename _Period::type >

6.316.1 Detailed Description

```
template<typename _Rep, typename _Period>
struct std::common_type< chrono::duration<_Rep, _Period>, chrono::duration<_Rep, _Period>>
```

Specialization of `common_type` for two identical `chrono::duration` types.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.317 `std::common_type< chrono::duration<_Rep1, _Period1>, chrono::duration<_Rep2, _Period2>>` Struct Template Reference

```
#include <chrono.h>
```

Inherits `__duration_common_type< common_type<_Rep1, _Rep2>, _Period1::type, _Period2::type>`.

6.317.1 Detailed Description

```
template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2>
struct std::common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > >
```

Specialization of `common_type` for `chrono::duration` types.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.318 `std::common_type< chrono::time_point< _Clock, _Duration > >` Struct Template Reference

```
#include <chrono.h>
```

Public Types

- using `type` = [chrono::time_point< _Clock, _Duration >](#)

6.318.1 Detailed Description

```
template<typename _Clock, typename _Duration>
struct std::common_type< chrono::time_point< _Clock, _Duration > >
```

Specialization of `common_type` for one `chrono::time_point` type.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.319 `std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >` Struct Template Reference

```
#include <chrono.h>
```

Public Types

- using `type` = [chrono::time_point< _Clock, _Duration >](#)

6.319.1 Detailed Description

```
template<typename _Clock, typename _Duration>
struct std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >
```

Specialization of `common_type` for two identical `chrono::time_point` types.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.320 `std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >` Struct Template Reference

```
#include <chrono.h>
```

Inherits `__timepoint_common_type< common_type< _Duration1, _Duration2 >, _Clock >`.

6.320.1 Detailed Description

```
template<typename _Clock, typename _Duration1, typename _Duration2>
struct std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >
```

Specialization of common_type for chrono::time_point types.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.321 std::compare_three_way_result< _Tp, _Up > Struct Template Reference

Inherits std::__detail::__cmp3way_res_impl< _Tp, _Up >.

6.321.1 Detailed Description

```
template<typename _Tp, typename _Up = _Tp>
struct std::compare_three_way_result< _Tp, _Up >
```

[cmp.result], result of three-way comparison

The documentation for this struct was generated from the following file:

- [compare](#)

6.322 std::complex< _Tp > Class Template Reference

Public Types

- typedef _Tp [value_type](#)

Public Member Functions

- constexpr [complex](#) (const _Tp &__r=_Tp(), const _Tp &__i=_Tp())
- constexpr **complex** (const [complex](#) &)=default
- template<typename _Up >
constexpr [complex](#) (const [complex](#)< _Up > &__z)
- constexpr [complex](#) __rep () const
- _GLIBCXX_ABI_TAG_CXX11 constexpr _Tp **imag** () const
- constexpr void **imag** (_Tp __val)
- constexpr [complex](#)< _Tp > & [operator*=](#) (const _Tp &)
- template<typename _Up >
constexpr [complex](#)< _Tp > & [operator*=](#) (const [complex](#)< _Up > &)
- constexpr [complex](#)< _Tp > & [operator+=](#) (const _Tp &__t)
- template<typename _Up >
constexpr [complex](#)< _Tp > & [operator+=](#) (const [complex](#)< _Up > &)
- constexpr [complex](#)< _Tp > & [operator-=](#) (const _Tp &__t)
- template<typename _Up >
constexpr [complex](#)< _Tp > & [operator-=](#) (const [complex](#)< _Up > &)
- constexpr [complex](#)< _Tp > & [operator/=](#) (const _Tp &)
- template<typename _Up >
constexpr [complex](#)< _Tp > & [operator/=](#) (const [complex](#)< _Up > &)
- constexpr [complex](#)< _Tp > & [operator=](#) (const _Tp &)
- constexpr [complex](#) & **operator=** (const [complex](#) &)=default
- template<typename _Up >
constexpr [complex](#)< _Tp > & [operator=](#) (const [complex](#)< _Up > &)
- _GLIBCXX_ABI_TAG_CXX11 constexpr _Tp **real** () const
- constexpr void **real** (_Tp __val)

6.322.1 Detailed Description

template<typename _Tp>
class std::complex<_Tp>

Template to represent complex numbers.

Specializations for float, double, and long double are part of the library. Results with any other type are not guaranteed.

Parameters

<i>Tp</i>	Type of real and imaginary values.
-----------	------------------------------------

6.322.2 Member Typedef Documentation

value_type

```
template<typename _Tp>
typedef _Tp std::complex<_Tp>::value_type
Value typedef.
```

6.322.3 Constructor & Destructor Documentation

complex() [1/2]

```
template<typename _Tp>
constexpr std::complex<_Tp>::complex (
    const _Tp & __r = _Tp(),
    const _Tp & __i = _Tp() ) [inline], [constexpr]
```

Default constructor. First parameter is x, second parameter is y. Unspecified parameters default to 0.

complex() [2/2]

```
template<typename _Tp>
template<typename _Up>
constexpr std::complex<_Tp>::complex (
    const complex<_Up> & __z ) [inline], [constexpr]
```

Converting constructor.

6.322.4 Member Function Documentation

operator+=()

```
template<typename _Tp>
constexpr complex<_Tp> & std::complex<_Tp>::operator+= (
    const _Tp & __t ) [inline], [constexpr]
```

Add a scalar to this complex number.

operator-=()

```
template<typename _Tp>
constexpr complex<_Tp> & std::complex<_Tp>::operator-= (
    const _Tp & __t ) [inline], [constexpr]
```

Subtract a scalar from this complex number.

The documentation for this class was generated from the following file:

- [complex](#)

6.323 `std::complex< double >` Class Reference

Public Types

- typedef `__complex__ double` `_ComplexT`
- typedef `double` `value_type`

Public Member Functions

- constexpr `complex` (`_ComplexT __z`)
- constexpr `complex` (`const complex< float > &__z`)
- constexpr `complex` (`const complex< long double > &`)
- constexpr `complex` (`double __r=0.0, double __i=0.0`)
- `__attribute__((abi_tag("cxx11")))` `const expr double imag()` `const`
- `__attribute__((abi_tag("cxx11")))` `const expr double real()` `const`
- constexpr `_ComplexT __rep` () `const`
- constexpr `void imag` (`double __val`)
- template<typename `_Tp` >
constexpr `complex & operator*=` (`const complex< _Tp > &__z`)
- constexpr `complex & operator*=` (`double __d`)
- template<typename `_Tp` >
constexpr `complex & operator+=` (`const complex< _Tp > &__z`)
- constexpr `complex & operator+=` (`double __d`)
- template<typename `_Tp` >
constexpr `complex & operator-=` (`const complex< _Tp > &__z`)
- constexpr `complex & operator-=` (`double __d`)
- template<typename `_Tp` >
constexpr `complex & operator/=` (`const complex< _Tp > &__z`)
- constexpr `complex & operator/=` (`double __d`)
- constexpr `complex & operator=` (`const complex &`)=default
- template<typename `_Tp` >
constexpr `complex & operator=` (`const complex< _Tp > &__z`)
- constexpr `complex & operator=` (`double __d`)
- constexpr `void real` (`double __val`)

6.323.1 Detailed Description

26.2.3 complex specializations `complex<double>` specialization

The documentation for this class was generated from the following file:

- [complex](#)

6.324 `std::complex< float >` Class Reference

Public Types

- typedef `__complex__ float` `_ComplexT`
- typedef `float` `value_type`

Public Member Functions

- constexpr **complex** (_ComplexT __z)
- constexpr **complex** (const **complex**< double > &)
- constexpr **complex** (const **complex**< long double > &)
- constexpr **complex** (float __r=0.0f, float __i=0.0f)
- **__attribute** ((__abi_tag__("cxx11"))) const expr float imag() const
- **__attribute** ((__abi_tag__("cxx11"))) const expr float real() const
- constexpr _ComplexT **__rep** () const
- constexpr void **imag** (float __val)
- template<class _Tp >
constexpr **complex** & **operator*=** (const **complex**< _Tp > &__z)
- constexpr **complex** & **operator*=** (float __f)
- template<typename _Tp >
constexpr **complex** & **operator+=** (const **complex**< _Tp > &__z)
- constexpr **complex** & **operator+=** (float __f)
- template<class _Tp >
constexpr **complex** & **operator-=** (const **complex**< _Tp > &__z)
- constexpr **complex** & **operator-=** (float __f)
- template<class _Tp >
constexpr **complex** & **operator/=** (const **complex**< _Tp > &__z)
- constexpr **complex** & **operator/=** (float __f)
- constexpr **complex** & **operator=** (const **complex** &)=default
- template<typename _Tp >
constexpr **complex** & **operator=** (const **complex**< _Tp > &__z)
- constexpr **complex** & **operator=** (float __f)
- constexpr void **real** (float __val)

6.324.1 Detailed Description

26.2.3 complex specializations complex<float> specialization

The documentation for this class was generated from the following file:

- [complex](#)

6.325 std::complex< long double > Class Reference**Public Types**

- typedef __complex__ long double **_ComplexT**
- typedef long double **value_type**

Public Member Functions

- constexpr **complex** (_ComplexT __z)
- constexpr **complex** (const **complex**< double > &__z)
- constexpr **complex** (const **complex**< float > &__z)
- constexpr **complex** (long double __r=0.0L, long double __i=0.0L)
- **__attribute** ((__abi_tag__("cxx11"))) const expr long double imag() const
- **__attribute** ((__abi_tag__("cxx11"))) const expr long double real() const
- constexpr _ComplexT **__rep** () const
- constexpr void **imag** (long double __val)
- template<typename _Tp >
constexpr **complex** & **operator*=** (const **complex**< _Tp > &__z)

- constexpr [complex](#) & **operator***= (long double __r)
- template<typename _Tp >
constexpr [complex](#) & **operator**+= (const [complex](#)<_Tp> &__z)
- constexpr [complex](#) & **operator**+= (long double __r)
- template<typename _Tp >
constexpr [complex](#) & **operator**-= (const [complex](#)<_Tp> &__z)
- constexpr [complex](#) & **operator**-= (long double __r)
- template<typename _Tp >
constexpr [complex](#) & **operator**/= (const [complex](#)<_Tp> &__z)
- constexpr [complex](#) & **operator**/= (long double __r)
- constexpr [complex](#) & **operator**= (const [complex](#) &)=default
- template<typename _Tp >
constexpr [complex](#) & **operator**= (const [complex](#)<_Tp> &__z)
- constexpr [complex](#) & **operator**= (long double __r)
- constexpr void **real** (long double __val)

6.325.1 Detailed Description

26.2.3 complex specializations [complex](#)<long double> specialization

The documentation for this class was generated from the following file:

- [complex](#)

6.326 `__gnu_pbds::detail::cond_dealtor< Entry, _Alloc >` Class Template Reference

```
#include <cond_key_dtor_entry_dealtor.hpp>
```

Public Types

- typedef HT_Map::entry **entry**
- typedef HT_Map::entry_allocator **entry_allocator**
- typedef alloc_traits::allocator_type **entry_allocator**
- typedef alloc_traits::pointer **entry_pointer**
- typedef HT_Map::key_type **key_type**

Public Member Functions

- **cond_dealtor** (entry_allocator *p_a, entry *p_e)
- **cond_dealtor** (entry_pointer p_e)
- void **set_key_destruct** ()
- void **set_no_action** ()
- void **set_no_action_destructor** ()

Protected Attributes

- bool **m_key_destruct**
- entry_allocator *const **m_p_a**
- entry *const **m_p_e**

6.326.1 Detailed Description

```
template<typename Entry, typename _Alloc>
class __gnu_pbds::detail::cond_dealtor< Entry, _Alloc >
```

Conditional deallocate constructor argument.

Conditional dey destructor, cc_hash.

The documentation for this class was generated from the following files:

- [cond_dealtor.hpp](#)
- [cond_key_dtor_entry_dealtor.hpp](#)

6.327 `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >` Class Template Reference

```
#include <ov_tree_map_.hpp>
```

Public Member Functions

- `cond_dtor` (value_vector a_vec, iterator &r_last_it, Size_Type total_size)
- void `set_no_action` ()

Protected Attributes

- value_vector `m_a_vec`
- const Size_Type `m_max_size`
- bool `m_no_action`
- iterator & `m_r_last_it`

6.327.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _↵
_Alloc>
```

```
template<typename Size_Type>
```

```
class __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor<
Size_Type >
```

Conditional destructor.

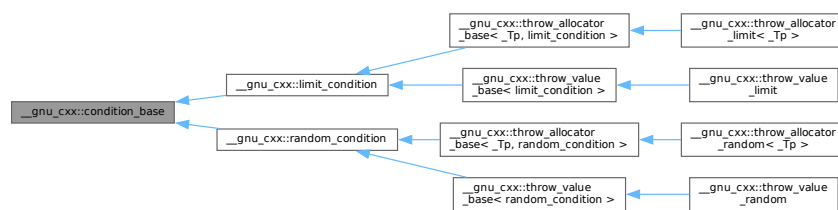
The documentation for this class was generated from the following file:

- [ov_tree_map_.hpp](#)

6.328 `__gnu_cxx::condition_base` Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for `__gnu_cxx::condition_base`:



Public Member Functions

- **condition_base** (const [condition_base](#) &)=default
- **condition_base & operator=** (const [condition_base](#) &)=default

6.328.1 Detailed Description

Base struct for condition policy.

Requires a public member function with the signature void throw_conditionally()

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.329 std::condition_variable Class Reference

Public Types

- typedef `__gthread_cond_t * native_handle_type`

Public Member Functions

- **condition_variable** (const [condition_variable](#) &)=delete
- native_handle_type **native_handle** ()
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- **condition_variable & operator=** (const [condition_variable](#) &)=delete
- void **wait** ([unique_lock](#)< [mutex](#) > &__lock)
- template<typename _Predicate >
void **wait** ([unique_lock](#)< [mutex](#) > &__lock, _Predicate __p)
- template<typename _Rep, typename _Period >
[cv_status](#) **wait_for** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime)
- template<typename _Rep, typename _Period, typename _Predicate >
bool **wait_for** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime, _Predicate __p)
- template<typename _Clock, typename _Duration >
[cv_status](#) **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime)
- template<typename _Clock, typename _Duration, typename _Predicate >
bool **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime, _Predicate __p)
- template<typename _Duration >
[cv_status](#) **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< [system_clock](#), _Duration > &__atime)

6.329.1 Detailed Description

`condition_variable`

The documentation for this class was generated from the following file:

- [condition_variable](#)

6.330 std::_V2::condition_variable_any Class Reference

Public Member Functions

- **condition_variable_any** (const [condition_variable_any](#) &)=delete
- void **notify_all** () noexcept

- `void notify_one () noexcept`
- `condition_variable_any & operator= (const condition_variable_any &)=delete`
- `template<typename _Lock >
void wait (_Lock &__lock)`
- `template<typename _Lock, typename _Predicate >
void wait (_Lock &__lock, _Predicate __p)`
- `template<class _Lock, class _Predicate >
bool wait (_Lock &__lock, stop_token __stoken, _Predicate __p)`
- `template<typename _Lock, typename _Rep, typename _Period >
cv_status wait_for (_Lock &__lock, const chrono::duration< _Rep, _Period > &__rtime)`
- `template<typename _Lock, typename _Rep, typename _Period, typename _Predicate >
bool wait_for (_Lock &__lock, const chrono::duration< _Rep, _Period > &__rtime, _Predicate __p)`
- `template<class _Lock, class _Rep, class _Period, class _Predicate >
bool wait_for (_Lock &__lock, stop_token __stoken, const chrono::duration< _Rep, _Period > &__rel_time, _Predicate __p)`
- `template<typename _Lock, typename _Clock, typename _Duration >
cv_status wait_until (_Lock &__lock, const chrono::time_point< _Clock, _Duration > &__atime)`
- `template<typename _Lock, typename _Clock, typename _Duration, typename _Predicate >
bool wait_until (_Lock &__lock, const chrono::time_point< _Clock, _Duration > &__atime, _Predicate __p)`
- `template<class _Lock, class _Clock, class _Duration, class _Predicate >
bool wait_until (_Lock &__lock, stop_token __stoken, const chrono::time_point< _Clock, _Duration > &__abs_time, _Predicate __p)`

6.330.1 Detailed Description

`condition_variable_any`

The documentation for this class was generated from the following file:

- [condition_variable](#)

6.331 `std::conditional< _Cond, _Iftrue, _Iffalse >` Struct Template Reference

Public Types

- `typedef _Iftrue type`

6.331.1 Detailed Description

`template<bool _Cond, typename _Iftrue, typename _Iffalse>`

`struct std::conditional< _Cond, _Iftrue, _Iffalse >`

Define a member typedef `type` to one of two argument types.

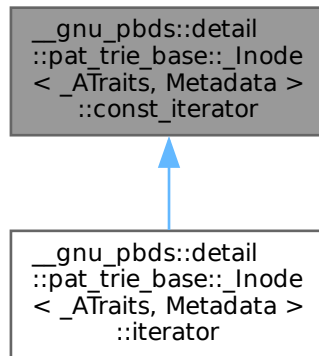
The documentation for this struct was generated from the following file:

- [type_traits](#)

6.332 `__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator` Struct Reference

`#include <pat_trie_base.hpp>`

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>::const_iterator`:



Public Types

- typedef `_Alloc::difference_type` **difference_type**
- typedef `std::forward_iterator_tag` **iterator_category**
- typedef `node_pointer_pointer` **pointer**
- typedef `node_pointer_reference` **reference**
- typedef `node_pointer` **value_type**

Public Member Functions

- **const_iterator** (`node_pointer_pointer p_p_cur=0, node_pointer_pointer p_p_end=0`)
- **bool operator!=** (`const const_iterator &other`) `const`
- `node_const_pointer` **operator*** () `const`
- `const_iterator &` **operator++** ()
- `const_iterator` **operator++** (`int`)
- `const node_pointer_pointer` **operator->** () `const`
- **bool operator==** (`const const_iterator &other`) `const`

Public Attributes

- `node_pointer_pointer` **m_p_p_cur**
- `node_pointer_pointer` **m_p_p_end**

6.332.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>::const_iterator
```

Constant child iterator.

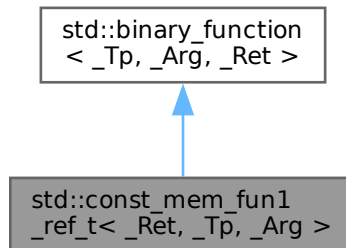
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.333 std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >:

**Public Types**

- typedef `_Tp` [first_argument_type](#)
- typedef `_Ret` [result_type](#)
- typedef `_Arg` [second_argument_type](#)

Public Member Functions

- `const_mem_fun1_ref_t` (`_Ret` (`_Tp::* __pf`) (`_Arg`) `const`)
- `_Ret operator()` (`const _Tp &__r, _Arg __x`) `const`

6.333.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

6.333.2 Member Typedef Documentation**first_argument_type**

```
typedef _Tp std::binary_function< _Tp , _Arg , _Ret >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef _Ret std::binary_function< _Tp , _Arg , _Ret >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Arg std::binary_function< _Tp , _Arg , _Ret >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

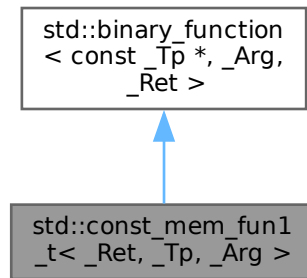
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.334 std::const_mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::const_mem_fun1_t< _Ret, _Tp, _Arg >:



Public Types

- typedef const _Tp * [first_argument_type](#)
- typedef _Ret [result_type](#)
- typedef _Arg [second_argument_type](#)

Public Member Functions

- `const_mem_fun1_t` (_Ret(_Tp::* __pf)(_Arg) const)
- `_Ret operator()` (const _Tp * __p, _Arg __x) const

6.334.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::const_mem_fun1_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

6.334.2 Member Typedef Documentation

first_argument_type

```
typedef const _Tp * std::binary\_function< const _Tp * , _Arg , _Ret >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef _Ret std::binary\_function< const _Tp * , _Arg , _Ret >::result_type [inherited]
result_type is the return type
```

second_argument_type

typedef _Arg [std::binary_function](#)< const _Tp *, _Arg , _Ret >::second_argument_type [inherited]
 second_argument_type is the type of the second argument

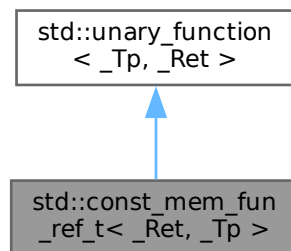
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.335 std::const_mem_fun_ref_t< _Ret, _Tp > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::const_mem_fun_ref_t< _Ret, _Tp >:

**Public Types**

- typedef _Tp [argument_type](#)
- typedef _Ret [result_type](#)

Public Member Functions

- **const_mem_fun_ref_t** (_Ret(_Tp::*__pf)() const)
- _Ret **operator()** (const _Tp &__r) const

6.335.1 Detailed Description

```
template<typename _Ret, typename _Tp>
class std::const_mem_fun_ref_t< _Ret, _Tp >
```

One of the [adaptors for member pointers](#).

6.335.2 Member Typedef Documentation**argument_type**

typedef _Tp [std::unary_function](#)< _Tp , _Ret >::argument_type [inherited]
 argument_type is the type of the argument

result_type

```
typedef _Ret std::unary_function< _Tp , _Ret >::result_type [inherited]
```

result_type is the return type

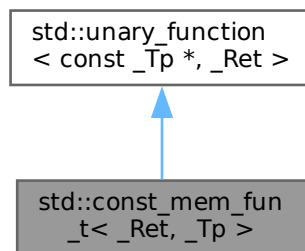
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.336 std::const_mem_fun_t< _Ret, _Tp > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::const_mem_fun_t< _Ret, _Tp >:

**Public Types**

- typedef const _Tp * [argument_type](#)
- typedef _Ret [result_type](#)

Public Member Functions

- **const_mem_fun_t** (_Ret(_Tp::*__pf)() const)
- _Ret **operator()** (const _Tp *__p) const

6.336.1 Detailed Description

```
template<typename _Ret, typename _Tp>
class std::const_mem_fun_t< _Ret, _Tp >
```

One of the [adaptors for member pointers](#).

6.336.2 Member Typedef Documentation**argument_type**

```
typedef const _Tp * std::unary_function< const _Tp * , _Ret >::argument_type [inherited]
```

argument_type is the type of the argument

result_type

```
typedef _Ret std::unary_function< const _Tp * , _Ret >::result_type [inherited]
```

`result_type` is the return type

The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.337 `__gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::Constant_binary_fun<_Result, _Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` **first_argument_type**
- typedef `_Result` **result_type**
- typedef `_Arg2` **second_argument_type**

Public Member Functions

- **constant_binary_fun** (const `_Result` &__v)
- const `result_type` & **operator()** (const `_Arg1` &, const `_Arg2` &) const

Public Attributes

- `_Result` **M_val**

6.337.1 Detailed Description

```
template<class _Result, class _Arg1 = _Result, class _Arg2 = _Arg1>
struct __gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2>
```

An [SGI extension](#) .

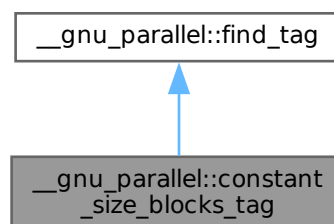
The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.338 `__gnu_parallel::constant_size_blocks_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::constant_size_blocks_tag`:



6.338.1 Detailed Description

Selects the constant block size variant for `std::find()`.

See also

`_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.339 `__gnu_cxx::constant_unary_fun<_Result, _Argument >` Struct Template Reference

Inherits `__gnu_cxx::Constant_unary_fun<_Result, _Argument >`.

Public Types

- typedef `_Argument` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- **constant_unary_fun** (const `_Result` &__v)
- const `result_type` & **operator()** (const `_Argument` &) const

Public Attributes

- `result_type` **_M_val**

6.339.1 Detailed Description

```
template<class _Result, class _Argument = _Result>
struct __gnu_cxx::constant_unary_fun<_Result, _Argument >
```

An [SGI extension](#) .

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.340 `__gnu_cxx::constant_void_fun<_Result >` Struct Template Reference

Inherits `__gnu_cxx::Constant_void_fun<_Result >`.

Public Types

- typedef `_Result` **result_type**

Public Member Functions

- **constant_void_fun** (const `_Result` &__v)
- const `result_type` & **operator()** () const

Public Attributes

- `result_type` **_M_val**

6.340.1 Detailed Description

```
template<class _Result>
struct __gnu_cxx::constant_void_fun< _Result >
```

An [SGI extension](#) .

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.341 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_Tl >` Struct Template Reference

6.341.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Tag, typename Policy_Tl = null_type>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_Tl >
```

Dispatch mechanism, primary template for associative types.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.342 `__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >` Struct Template Reference

```
#include <priority_queue_base_dispatch.hpp>
```

Public Types

- typedef [binary_heap](#)< _VTp, Cmp_Fn, _Alloc > [type](#)

6.342.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >
```

Specialization for `binary_heap`.

6.342.2 Member Typedef Documentation

type

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef binary\_heap<_VTp, Cmp_Fn, _Alloc> \_\_gnu\_pbds::detail::container\_base\_dispatch< _VTp,
Cmp_Fn, _Alloc, binary\_heap\_tag, null\_type >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

6.343 `__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >` Struct Template Reference

```
#include <priority_queue_base_dispatch.hpp>
```

Public Types

- typedef [binomial_heap](#)< _VTp, Cmp_Fn, _Alloc > [type](#)

6.343.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >
```

Specialization for binomial_heap.

6.343.2 Member Typedef Documentation

type

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef binomial_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch< _VTp,
Cmp_Fn, _Alloc, binomial_heap_tag, null_type >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

6.344 __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type > Struct Template Reference

```
#include <priority_queue_base_dispatch.hpp>
```

Public Types

- typedef [pairing_heap](#)< _VTp, Cmp_Fn, _Alloc > [type](#)

6.344.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type >
```

Specialization for pairing_heap.

6.344.2 Member Typedef Documentation

type

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef pairing_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch< _VTp,
Cmp_Fn, _Alloc, pairing_heap_tag, null_type >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

6.345 __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type > Struct Template Reference

```
#include <priority_queue_base_dispatch.hpp>
```

Public Types

- typedef [rc_binomial_heap](#)< _VTp, Cmp_Fn, _Alloc > [type](#)

6.345.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type>
```

Specialization for `rc_binary_heap`.

6.345.2 Member Typedef Documentation

type

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef rc_binomial_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch<_VTp,
Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

6.346 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>` Struct Template Reference

```
#include <priority_queue_base_dispatch.hpp>
```

Public Types

- typedef `thin_heap<_VTp, Cmp_Fn, _Alloc>` [type](#)

6.346.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>
```

Specialization for `thin_heap`.

6.346.2 Member Typedef Documentation

type

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef thin_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

6.347 `__gnu_pbds::detail::container_base_dispatch<Key, Mapped, _Alloc, cc_hash_tag, Policy_Tl>` Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef `cc_ht_map<Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at2t>` [type](#)

6.347.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_Tl >
```

Specialization collision-chaining hash map.

6.347.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef cc_ht_map<Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at2t> __gnu_pbds::detail::container_base_d
Key, Mapped, _Alloc, cc_hash_tag, Policy_Tl >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.348 __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [gp_ht_map](#)< Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t > [type](#)

6.348.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl >
```

Specialization general-probe hash map.

6.348.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef gp_ht_map<Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t> __gnu_pbds::detail::container_k
Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.349 __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_Tl > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [lu_map](#)< Key, Mapped, at0t, _Alloc, at1t > [type](#)

6.349.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_Tl >
```

Specialization for list-update map.

6.349.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef lu_map<Key, Mapped, at0t, _Alloc, at1t> __gnu_pbds::detail::container_base_dispatch< Key,
Mapped, _Alloc, list_update_tag, Policy_Tl >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.350 **__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl > Struct Template Reference**

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [ov_tree_map](#)< Key, Mapped, at0t, at1t, _Alloc > **type**

6.350.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl >
```

Specialization ordered-vector tree map.

6.350.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef ov_tree_map<Key, Mapped, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch<
Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl >::type
Dispatched type.
```

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.351 **__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_Tl > Struct Template Reference**

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [pat_trie_map](#)< Key, Mapped, at1t, _Alloc > **type**

6.351.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_Tl >
```

Specialization for PATRICIA trie map.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.352 __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_Tl > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [rb_tree_map](#)< Key, Mapped, at0t, at1t, _Alloc > [type](#)

6.352.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_Tl >
```

Specialization for R-B tree map.

6.352.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef rb\_tree\_map<Key, Mapped, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch<
Key, Mapped, _Alloc, rb\_tree\_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.353 __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_Tl > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef [splay_tree_map](#)< Key, Mapped, at0t, at1t, _Alloc > [type](#)

6.353.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_Tl >
```

Specialization splay tree map.

6.353.2 Member Typedef Documentation

type

```
template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl >
typedef splay\_tree\_map<Key, Mapped, at0t, at1t, _Alloc> \_\_gnu\_pbds::detail::container\_base\_dispatch<
Key, Mapped, _Alloc, splay\_tree\_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.354 **__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_Tl > Struct Template Reference**

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef cc_ht_set< Key, [null_type](#), at0t, at1t, _Alloc, at3t::value, at4t, at2t > [type](#)

6.354.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct \_\_gnu\_pbds::detail::container\_base\_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_Tl >
```

Specialization colision-chaining hash set.

6.354.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef cc_ht_set<Key, null\_type, at0t, at1t, _Alloc, at3t::value, at4t, at2t> \_\_gnu\_pbds::detail::container\_base\_dispatch<
Key, null\_type, _Alloc, cc\_hash\_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.355 **__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_Tl > Struct Template Reference**

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef gp_ht_set< Key, [null_type](#), at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t > [type](#)

6.355.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct \_\_gnu\_pbds::detail::container\_base\_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_Tl >
```

Specialization general-probe hash set.

6.355.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef gp_ht_set<Key, null\_type, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t> \_\_gnu\_pbds::detail::container\_base\_dispatch<
Key, null\_type, _Alloc, gp\_hash\_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.356 [__gnu_pbds::detail::container_base_dispatch](#)< Key, [null_type](#), [_Alloc](#), [list_update_tag](#), Policy_Tl > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef lu_set< Key, [null_type](#), at0t, [_Alloc](#), at1t > [type](#)

6.356.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct \_\_gnu\_pbds::detail::container\_base\_dispatch< Key, null\_type, \_Alloc, list\_update\_tag, Policy_TI >
```

Specialization for list-update set.

6.356.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef lu_set<Key, null\_type, at0t, \_Alloc, at1t> \_\_gnu\_pbds::detail::container\_base\_dispatch<
Key, null\_type, \_Alloc, list\_update\_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.357 [__gnu_pbds::detail::container_base_dispatch](#)< Key, [null_type](#), [_Alloc](#), [ov_tree_tag](#), Policy_TI > Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef ov_tree_set< Key, [null_type](#), at0t, at1t, [_Alloc](#) > [type](#)

6.357.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct \_\_gnu\_pbds::detail::container\_base\_dispatch< Key, null\_type, \_Alloc, ov\_tree\_tag, Policy_TI >
```

Specialization ordered-vector tree set.

6.357.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef ov_tree_set<Key, null_type, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch<
Key, null_type, _Alloc, ov_tree_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.358 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_Tl >` Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef `pat_trie_set< Key, null_type, at1t, _Alloc >` `type`

6.358.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_Tl >
```

Specialization for PATRICIA trie set.

6.358.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef pat_trie_set<Key, null_type, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch<
Key, null_type, _Alloc, pat_trie_tag, Policy_Tl >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.359 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_Tl >` Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef `rb_tree_set< Key, null_type, at0t, at1t, _Alloc >` `type`

6.359.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_Tl >
```

Specialization for R-B tree set.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.360 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >` Struct Template Reference

```
#include <container_base_dispatch.hpp>
```

Public Types

- typedef `splay_tree_set< Key, null_type, at0t, at1t, _Alloc >` `type`

6.360.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >
```

Specialization splay tree set.

6.360.2 Member Typedef Documentation

type

```
template<typename Key , typename _Alloc , typename Policy_Tl >
typedef splay_tree_set<Key, null_type, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch<
Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >::type
Dispatched type.
```

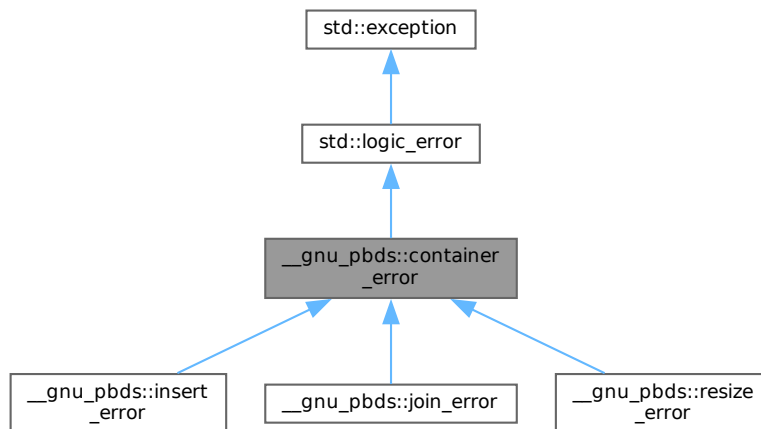
The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

6.361 `__gnu_pbds::container_error` Struct Reference

```
#include <exception.hpp>
```

Inheritance diagram for `__gnu_pbds::container_error`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.361.1 Detailed Description

Base class for exceptions.

6.361.2 Member Function Documentation

`what()`

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

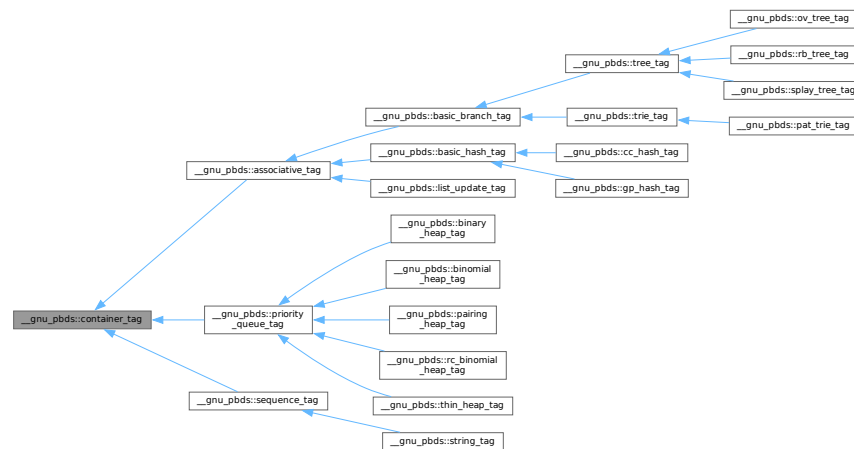
The documentation for this struct was generated from the following file:

- [exception.hpp](#)

6.362 `__gnu_pbds::container_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::container_tag`:



6.362.1 Detailed Description

Base data structure tag.

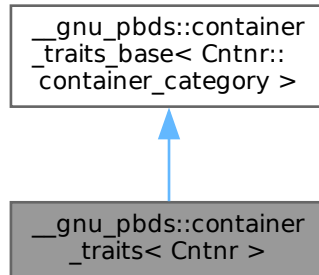
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.363 `__gnu_pbds::container_traits< Cntnr >` Struct Template Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::container_traits< Cntnr >`:



Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `container_traits_base< container_category >` **base_type**
- typedef `Cntnr::container_category` **container_category**
- typedef `Cntnr` **container_type**
- typedef `base_type::invalidation_guarantee` **invalidation_guarantee**

6.363.1 Detailed Description

```
template<typename Cntnr>
struct __gnu_pbds::container_traits< Cntnr >
```

Container traits.

6.363.2 Member Enumeration Documentation

anonymous enum

```
template<typename Cntnr >
anonymous enum
```

Enumerator

<code>order_preserving</code>	True only if Cntnr objects guarantee storing keys by order.
<code>erase_can_throw</code>	True only if erasing a key can throw.
<code>split_join_can_throw</code>	True only if split or join operations can throw.
<code>reverse_iteration</code>	True only reverse iterators are supported.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.364 `__gnu_pbds::container_traits_base<_Tag>` Struct Template Reference

6.364.1 Detailed Description

`template<typename _Tag>`

`struct __gnu_pbds::container_traits_base<_Tag>`

Primary template, container traits base.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.365 `__gnu_pbds::container_traits_base<binary_heap_tag>` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `binary_heap_tag` `container_category`
- typedef `basic_invalidation_guarantee` `invalidation_guarantee`

6.365.1 Detailed Description

Specialization, binary heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.366 `__gnu_pbds::container_traits_base<binomial_heap_tag>` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `binomial_heap_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

6.366.1 Detailed Description

Specialization, binomial heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.367 `__gnu_pbds::container_traits_base<cc_hash_tag>` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `cc_hash_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

6.367.1 Detailed Description

Specialization, cc hash.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.368 `__gnu_pbds::container_traits_base< gp_hash_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `gp_hash_tag` `container_category`
- typedef `basic_invalidation_guarantee` `invalidation_guarantee`

6.368.1 Detailed Description

Specialization, gp hash.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.369 `__gnu_pbds::container_traits_base< list_update_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `list_update_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

6.369.1 Detailed Description

Specialization, list update.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.370 `__gnu_pbds::container_traits_base< ov_tree_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `ov_tree_tag` `container_category`
- typedef `basic_invalidation_guarantee` `invalidation_guarantee`

6.370.1 Detailed Description

Specialization, ov tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.371 `__gnu_pbds::container_traits_base< pairing_heap_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `pairing_heap_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

6.371.1 Detailed Description

Specialization, pairing heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.372 `__gnu_pbds::container_traits_base< pat_trie_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `pat_trie_tag` `container_category`
- typedef `range_invalidation_guarantee` `invalidation_guarantee`

6.372.1 Detailed Description

Specialization, pat trie.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.373 `__gnu_pbds::container_traits_base< rb_tree_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { `order_preserving` , `erase_can_throw` , `split_join_can_throw` , `reverse_iteration` }
- typedef `rb_tree_tag` `container_category`
- typedef `range_invalidation_guarantee` `invalidation_guarantee`

6.373.1 Detailed Description

Specialization, rb tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.374 `__gnu_pbds::container_traits_base< rc_binomial_heap_tag >` Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { **order_preserving** , **erase_can_throw** , **split_join_can_throw** , **reverse_iteration** }
- typedef [rc_binomial_heap_tag](#) **container_category**
- typedef [point_invalidation_guarantee](#) **invalidation_guarantee**

6.374.1 Detailed Description

Specialization, rc binomial heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.375 __gnu_pbds::container_traits_base< splay_tree_tag > Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { **order_preserving** , **erase_can_throw** , **split_join_can_throw** , **reverse_iteration** }
- typedef [splay_tree_tag](#) **container_category**
- typedef [range_invalidation_guarantee](#) **invalidation_guarantee**

6.375.1 Detailed Description

Specialization, splay tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.376 __gnu_pbds::container_traits_base< thin_heap_tag > Struct Reference

```
#include <tag_and_trait.hpp>
```

Public Types

- enum { **order_preserving** , **erase_can_throw** , **split_join_can_throw** , **reverse_iteration** }
- typedef [thin_heap_tag](#) **container_category**
- typedef [point_invalidation_guarantee](#) **invalidation_guarantee**

6.376.1 Detailed Description

Specialization, thin heap.

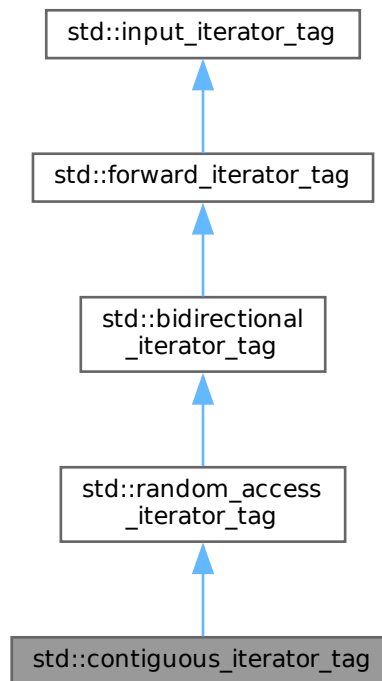
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.377 std::contiguous_iterator_tag Struct Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for std::contiguous_iterator_tag:



6.377.1 Detailed Description

Contiguous iterators point to objects stored contiguously in memory.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.378 std::counted_iterator<_It> Class Template Reference

```
#include <stl_iterator.h>
```

Inherits std::__detail::__counted_iter_value_type<_It>, std::__detail::__counted_iter_concept<_It>, and std::__detail::__counted_iter_cat<_It>.

Public Types

- using **difference_type** = iter_difference_t<_It>
- using **iterator_type** = _It

Public Member Functions

- constexpr **counted_iterator** (_It __i, iter_difference_t<_It> __n)
- template<typename _It2>
requires convertible_to<const _It2&, _It>
constexpr **counted_iterator** (const [counted_iterator](#)<_It2> &__x)

- constexpr **_It base** () &&noexcept(is_nothrow_move_constructible_v<_It >)
- constexpr const **_It & base** () const &noexcept
- constexpr iter_difference_t<_It > **count** () const noexcept
- constexpr decltype(auto) **operator*** () const noexcept(noexcept(*_M_current))
- constexpr decltype(auto) **operator*** () noexcept(noexcept(*_M_current))
- constexpr **counted_iterator operator+** (iter_difference_t<_It > __n) const
- constexpr **counted_iterator & operator++** ()
- constexpr decltype(auto) **operator++** (int)
- constexpr **counted_iterator operator++** (int)
- constexpr **counted_iterator & operator+=** (iter_difference_t<_It > __n)
- constexpr **counted_iterator operator-** (iter_difference_t<_It > __n) const
- constexpr **counted_iterator & operator--** ()
- constexpr **counted_iterator operator--** (int)
- constexpr **counted_iterator & operator-=** (iter_difference_t<_It > __n)
- constexpr auto **operator->** () const noexcept
- template<typename _It2 >
requires assignable_from<_It&, const _It2&>
constexpr **counted_iterator & operator=** (const **counted_iterator**<_It2 > &__x)
- constexpr decltype(auto) **operator[]** (iter_difference_t<_It > __n) const noexcept(noexcept(_M_current[__n]))

Friends

- constexpr iter_rvalue_reference_t<_It > **iter_move** (const **counted_iterator** &__i) noexcept(noexcept(ranges::iter_move(__i._M_current)))
- template<indirectly_swappable<_It > _It2>
constexpr void **iter_swap** (const **counted_iterator** &__x, const **counted_iterator**<_It2 > &__y) noexcept(noexcept(ranges::iter_swap(__x._M_current, __y._M_current)))
- constexpr **counted_iterator operator+** (iter_difference_t<_It > __n, const **counted_iterator** &__x)
- template<common_with<_It > _It2>
constexpr iter_difference_t<_It2 > **operator-** (const **counted_iterator** &__x, const **counted_iterator**<_It2 > &__y)
- constexpr iter_difference_t<_It > **operator-** (const **counted_iterator** &__x, default_sentinel_t)
- constexpr iter_difference_t<_It > **operator-** (default_sentinel_t, const **counted_iterator** &__y)
- template<common_with<_It > _It2>
constexpr strong_ordering **operator<=** (const **counted_iterator** &__x, const **counted_iterator**<_It2 > &__y)
- template<common_with<_It > _It2>
constexpr bool **operator==** (const **counted_iterator** &__x, const **counted_iterator**<_It2 > &__y)
- constexpr bool **operator==** (const **counted_iterator** &__x, default_sentinel_t)

6.378.1 Detailed Description

```
template<input_or_output_iterator _It>
class std::counted_iterator<_It >
```

An iterator adaptor that keeps track of the distance to the end.

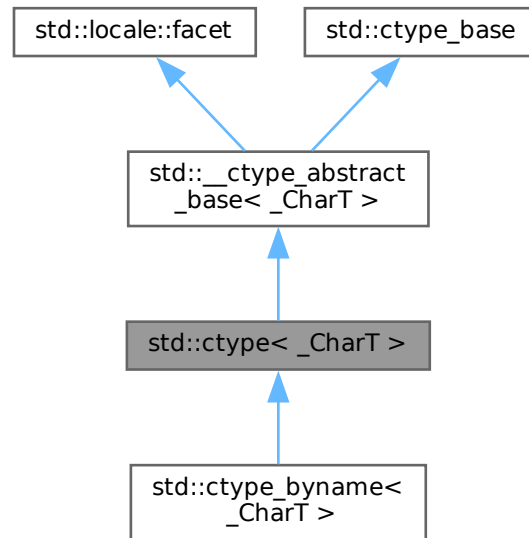
The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.379 std::ctype<_CharT> Class Template Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::ctype<_CharT>:



Public Types

- typedef const int * **__to_type**
- typedef _CharT **char_type**
- typedef `__ctype_abstract_base<_CharT>::mask` **mask**

Public Member Functions

- **ctype** (size_t __refs=0)
- const char_type * **is** (const char_type * __lo, const char_type * __hi, mask * __vec) const
- bool **is** (mask __m, char_type __c) const
- char **narrow** (char_type __c, char __dfault) const
- const char_type * **narrow** (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const
- const char_type * **scan_is** (mask __m, const char_type * __lo, const char_type * __hi) const
- const char_type * **scan_not** (mask __m, const char_type * __lo, const char_type * __hi) const
- const char_type * **tolower** (char_type * __lo, const char_type * __hi) const
- char_type **tolower** (char_type __c) const
- const char_type * **toupper** (char_type * __lo, const char_type * __hi) const
- char_type **toupper** (char_type __c) const
- char_type **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, char_type * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual const char_type * [do_is](#) (const char_type * __lo, const char_type * __hi, mask * __vec) const
- virtual bool [do_is](#) (mask __m, char_type __c) const
- virtual char [do_narrow](#) (char_type, char __default) const
- virtual const char_type * [do_narrow](#) (const char_type * __lo, const char_type * __hi, char __default, char * __to) const
- virtual const char_type * [do_scan_is](#) (mask __m, const char_type * __lo, const char_type * __hi) const
- virtual const char_type * [do_scan_not](#) (mask __m, const char_type * __lo, const char_type * __hi) const
- virtual const char_type * [do_tolower](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_tolower](#) (char_type __c) const
- virtual const char_type * [do_toupper](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_toupper](#) (char_type __c) const
- virtual char_type [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, char_type * __dest) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char * __s)

6.379.1 Detailed Description

template<typename _CharT>
class std::ctype<_CharT >

Primary class template ctype facet.

This template class defines classification and conversion functions for character sets. It wraps ctype functionality. Ctype gets used by streams for many I/O operations.

This template provides the protected virtual functions the developer will have to replace in a derived class or specialization to make a working facet. The public functions that access them are defined in `__ctype_abstract_base`, to allow for implementation flexibility. See `ctype<wchar_t>` for an example. The functions are documented in `__ctype_abstract_base`.

Note: implementations are provided for all the protected virtual functions, but will likely not be useful.

6.379.2 Member Function Documentation

do_is() [1/2]

```
template<typename _CharT >
virtual const char_type * std::ctype<_CharT>::do_is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [protected], [virtual]
```

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the input.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_is() [2/2]

```
template<typename _CharT >
virtual bool std::ctype<_CharT>::do_is (
    mask __m,
    char_type __c ) const [protected], [virtual]
```

Test char_type classification.

This function finds a mask M for c and compares it to mask m.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__c</code>	The char_type to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

$(M \& _m) \neq 0$.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_narrow() [1/2]

```
template<typename _CharT >
virtual char std::ctype<_CharT>::do_narrow (
```

```
char_type __c,
char __default ) const [protected], [virtual]
```

Narrow `char_type` to `char`.

This virtual function converts the argument to `char` using the simplest reasonable transformation. If the conversion fails, `__default` is returned instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted `char`.

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::narrow\(\)](#), and [std::ctype<char>::narrow\(\)](#).

`do_narrow()` [2/2]

```
template<typename _CharT >
virtual const char_type * std::ctype<_CharT>::do_narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [protected], [virtual]
```

Narrow `char_type` array to `char`.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__default` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

`do_scan_is()`

```
template<typename _CharT >
virtual const char_type * std::ctype<_CharT>::do_scan_is (
```

```
mask __m,
const char_type * __lo,
const char_type * __hi ) const [protected], [virtual]
```

Find char_type matching mask.

This function searches for and returns the first char_type c in [__lo,__hi) for which is(__m,c) is true.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char_type if found, else __hi.

Implements [std::ctype_abstract_base<_CharT>](#).

do_scan_not()

```
template<typename _CharT>
virtual const char_type * std::ctype<_CharT>::do_scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char_type if found, else __hi.

Implements [std::ctype_abstract_base<_CharT>](#).

do_tolower() [1/2]

```
template<typename _CharT>
virtual const char_type * std::ctype<_CharT>::do_tolower (
```

```
char_type * __lo,
const char_type * __hi ) const [protected], [virtual]
```

Convert array to lowercase.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

`do_tolower()` [2/2]

```
template<typename _CharT >
virtual char_type std::ctype<_CharT>::do_tolower (
    char_type __c ) const [protected], [virtual]
```

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The lowercase `char_type` if convertible, else `__c`.

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::tolower\(\)](#), and [std::ctype<char>::tolower\(\)](#).

`do_toupper()` [1/2]



```
template<typename _CharT >
virtual const char_type * std::ctype<_CharT>::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Convert array to uppercase.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

 <code>__lo</code>	Pointer to start of range.
 <code>__hi</code>	Pointer to end of range.

Returns

`__hi`.Implements [std::__ctype_abstract_base<_CharT>](#).**do_toupper()** [2/2]


```
template<typename _CharT >
virtual char_type std::ctype<_CharT>::do_toupper (
    char_type __c ) const [protected], [virtual]
```

Convert to uppercase.

This virtual function converts the `char_type` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

 <code>__c</code>	The <code>char_type</code> to convert.
---	--

Returns

The uppercase `char_type` if convertible, else `__c`.Implements [std::__ctype_abstract_base<_CharT>](#).Referenced by [std::ctype<char>::toupper\(\)](#), and [std::ctype<char>::toupper\(\)](#).**do_widen()** [1/2]

```
template<typename _CharT >
virtual char_type std::ctype<_CharT>::do_widen (
    char __c ) const [protected], [virtual]
```


Widen char.

This virtual function converts the `char` to `char_type` using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__c</code>	The char to convert.
---	----------------------

Returns

The converted `char_type`

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::widen\(\)](#), and [std::ctype<char>::widen\(\)](#).

do_widen() [2/2]

```
template<typename _CharT >
virtual const char * std::ctype<\_CharT>::do\_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [protected], [virtual]
```

Widen char array.

This function converts each char in the input to `char_type` using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

__lo	Pointer to start range.
__hi	Pointer to end of range.
__to	Pointer to the destination array.

Returns

[__hi](#).

Implements [std::__ctype_abstract_base<_CharT>](#).

is() [1/2]

```
template<typename _CharT >
const char_type * std::\_\_ctype\_abstract\_base<\_CharT>::is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [inline], [inherited]
```

Return a mask array.

This function finds the mask for each `char_type` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the char array. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

__lo	Pointer to start of range.
__hi	Pointer to end of range.
__vec	Pointer to an array of mask storage.

Returns

`__hi.`References [std::__ctype_abstract_base< _CharT >::do_is\(\)](#).**is()** [2/2]

```
template<typename _CharT >
bool std::__ctype_abstract_base< _CharT >::is (
    mask __m,
    char_type __c ) const [inline], [inherited]
```

Test char_type classification.

This function finds a mask M for `__c` and compares it to mask `__m`. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

<code>__c</code>	The char_type to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0.`References [std::__ctype_abstract_base< _CharT >::do_is\(\)](#).Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), and [std::time_get< _CharT, _InIter >::get\(\)](#).**narrow()** [1/2]

```
template<typename _CharT >
char std::__ctype_abstract_base< _CharT >::narrow (
    char_type __c,
    char __dfault ) const [inline], [inherited]
```

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. It does so by returning `ctype<char_type>::do_narrow(__c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), and [std::time_put<_CharT, _OutIter>::put\(\)](#).

narrow() [2/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline], [inherited]
```

Narrow array to char array.

This function converts each char_type in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char_type in the input that cannot be converted, *default* is used instead. It does so by returning `ctype<char_type>::do_narrow(__lo, __hi, __default, __to)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

scan_is()

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::scan_is (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Find char_type matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which `is(m,c)` is true. It does so by returning `ctype<char_type>::do_scan_is()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_is\(\)](#).




scan_not()

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

 __m	The mask to compare against.
 __lo	Pointer to first char in range.
 __hi	Pointer to end of range.

Returns

Pointer to non-matching char if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_not\(\)](#).



tolower() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to lowercase.

This function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_tolower(__lo,__hi).

Parameters

 __lo	Pointer to start of range.
 __hi	Pointer to end of range.

Returns

__hi.

References [std::__ctype_abstract_base<_CharT>::do_tolower\(\)](#).


tolower() [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base< _CharT >::tolower (
    char_type __c ) const [inline], [inherited]
```

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_tolower(c).

Parameters

 __c	The char_type to convert.
--	---------------------------

Returns

The lowercase char_type if convertible, else __c.

References [std::__ctype_abstract_base< _CharT >::do_tolower\(\)](#).

Referenced by [std::time_get< _CharT, _InIter >::get\(\)](#).



toupper() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base< _CharT >::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

 __lo	Pointer to start of range.
 __hi	Pointer to end of range.

Returns

__hi.

References [std::__ctype_abstract_base< _CharT >::do_toupper\(\)](#).


toupper() [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base< _CharT >::toupper (
    char_type __c ) const [inline], [inherited]
```

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_toupper(c).

Parameters

 <code>__c</code>	The char_type to convert.
--	---------------------------

Returns

The uppercase char_type if convertible, else `__c`.

References [std::__ctype_abstract_base<_CharT>::do_toupper\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get\(\)](#).

widen() [1/2]


```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT>::widen (
    char __c ) const [inline], [inherited]
```

Widen char to char_type.

This function converts the char argument to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__c</code>	The char to convert.
--	----------------------

Returns

The converted char_type.

References [std::__ctype_abstract_base<_CharT>::do_widen\(\)](#).

Referenced by [std::money_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), and [std::time_put<_CharT, _OutIter>::do_put\(\)](#).

widen() [2/2]




```
template<typename _CharT >
const char * std::__ctype_abstract_base<_CharT>::widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline], [inherited]
```

Widen array to char_type.

This function converts each char in the input to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__lo</code>	Pointer to start of range.
 <code>__hi</code>	Pointer to end of range.
 <code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_widen\(\)](#).

6.379.3 Member Data Documentation

id

```
template<typename _CharT >
locale::id std::ctype<_CharT >::id [static]
The facet id for ctype<char_type>
```

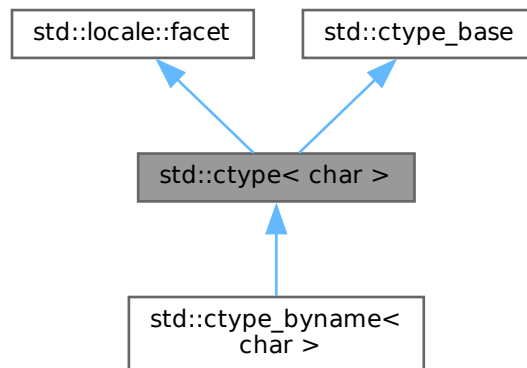
The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.380 std::ctype< char > Class Reference

```
#include <locale_facets.h>
```

Inheritance diagram for `std::ctype< char >`:



Public Types

- typedef const int * `__to_type`
- typedef char `char_type`
- typedef unsigned short `mask`

Public Member Functions

- `ctype` (`__c_locale __cloc`, const `mask * __table=0`, bool `__del=false`, size_t `__refs=0`)
- `ctype` (const `mask * __table=0`, bool `__del=false`, size_t `__refs=0`)
- const `char * is` (const `char * __lo`, const `char * __hi`, const `mask * __vec`) const
- bool `is` (mask `__m`, char `__c`) const
- char `narrow` (char_type `__c`, char `__dfault`) const
- const `char_type * narrow` (const `char_type * __lo`, const `char_type * __hi`, char `__dfault`, char * `__to`) const

- const char * [scan_is](#) (mask __m, const char * __lo, const char * __hi) const
- const char * [scan_not](#) (mask __m, const char * __lo, const char * __hi) const
- const mask * [table](#) () const throw ()
- const [char_type](#) * [tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [tolower](#) ([char_type](#) __c) const
- const [char_type](#) * [toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [toupper](#) ([char_type](#) __c) const
- [char_type](#) [widen](#) (char __c) const
- const char * [widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Public Member Functions

- static const mask * [classic_table](#) () throw ()

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const size_t [table_size](#)
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual [~ctype](#) ()
- virtual char [do_narrow](#) ([char_type](#) __c, char __default) const
- virtual const [char_type](#) * [do_narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, char __default, char * __to) const
- virtual const [char_type](#) * [do_tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_tolower](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_toupper](#) ([char_type](#) __c) const
- virtual [char_type](#) [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char * __s)

Protected Attributes

- `__c_locale __M_c_locale_ctype`
- `bool __M_del`
- `char __M_narrow [1+static_cast< unsigned char >(-1)]`
- `char __M_narrow_ok`
- `const mask * __M_table`
- `__to_type __M_tolower`
- `__to_type __M_toupper`
- `char __M_widen [1+static_cast< unsigned char >(-1)]`
- `char __M_widen_ok`

6.380.1 Detailed Description

The `ctype<char>` specialization.

This class defines classification and conversion functions for the `char` type. It gets used by `char` streams for many I/O operations. The `char` specialization provides a number of optimizations as well.

6.380.2 Member Typedef Documentation

`char_type`

```
typedef char std::ctype< char >::char_type
```

Typedef for the template parameter `char`.

6.380.3 Constructor & Destructor Documentation

`ctype()` [1/2]

```
std::ctype< char >::ctype (
    const mask * __table = 0,
    bool __del = false,
    size_t __refs = 0 ) [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__table</code>	If non-zero, table is used as the per-char mask. Else <code>classic_table()</code> is used.
<code>__del</code>	If true, passes ownership of table to this facet.
<code>__refs</code>	Passed to the base facet class.

`ctype()` [2/2]

```
std::ctype< char >::ctype (
    __c_locale __cloc,
    const mask * __table = 0,
    bool __del = false,
    size_t __refs = 0 ) [explicit]
```

Constructor performs static initialization.

This constructor is used to construct the initial C locale facet.

Parameters

<code>__cloc</code>	Handle to C locale data.
<code>__table</code>	If non-zero, table is used as the per-char mask.
<code>__del</code>	If true, passes ownership of table to this facet.
<code>__refs</code>	Passed to the base facet class.

`~ctype()`

```
virtual std::ctype< char >::~~ctype ( ) [protected], [virtual]
```

Destructor.

This function deletes `table()` if `del` was true in the constructor.

6.380.4 Member Function Documentation

`classic_table()`

```
static const mask * std::ctype< char >::classic_table ( ) throw ( ) [static]
```

Returns a pointer to the C locale mask table.

`do_narrow()` [1/2]

```
virtual char std::ctype< char >::do_narrow (
    char_type __c,
    char __default ) const [inline], [protected], [virtual]
```

Narrow char.

This virtual function converts the `char` to `char` using the simplest reasonable transformation. If the conversion fails, `default` is returned instead. For an underived `ctype<char>` facet, `c` will be returned unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

`do_narrow()` [2/2]

```
virtual const char_type * std::ctype< char >::do_narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline], [protected], [virtual]
```

Narrow char array to char array.

This virtual function converts each `char` in the range `[lo,hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any `char` in the input that cannot be converted, `default` is used instead. For an underived `ctype<char>` facet, the argument will be copied unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

`do_tolower()` [1/2]

```
virtual const char_type * std::ctype< char >::do_tolower (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Convert array to lowercase.

This virtual function converts each char in the range `[lo,hi)` to lowercase if possible. Other chars remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

`do_tolower()` [2/2]

```
virtual char_type std::ctype< char >::do_tolower (
    char_type __c ) const [protected], [virtual]
```

Convert to lowercase.

This virtual function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The lowercase char if convertible, else `__c`.

do_toupper() [1/2]

```
virtual const char_type * std::ctype< char >::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Convert array to uppercase.

This virtual function converts each char in the range [lo,hi) to uppercase if possible. Other chars remain untouched.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

do_toupper() [2/2]

```
virtual char_type std::ctype< char >::do_toupper (
    char_type __c ) const [protected], [virtual]
```

Convert to uppercase.

This virtual function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

do_widen() [1/2]

```
virtual char_type std::ctype< char >::do_widen (
    char __c ) const [inline], [protected], [virtual]
```

Widen char.

This virtual function converts the char to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be returned unchanged.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecv`t for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

`do_widen()` [2/2]

```
virtual const char * std::ctype< char >::do_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline], [protected], [virtual]
```

Widen char array.

This function converts each char in the range [lo,hi) to char using the simplest reasonable transformation. For an undervied `ctype<char>` facet, the argument will be copied unchanged.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecv`t for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

`is()` [1/2]

```
const char * std::ctype< char >::is (
    const char * __lo,
    const char * __hi,
    mask * __vec ) const [inline]
```

Return a mask array.

This function finds the mask for each char in the range [lo, hi) and successively writes it to `vec`. `vec` must have as many elements as the char array.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns`__hi`.**is()** [2/2]

```
bool std::ctype< char >::is (
    mask __m,
    char __c ) const [inline]
```

Test char classification.

This function compares the mask table[c] to `__m`.**Parameters**

<code>__c</code>	The char to compare the mask of.
<code>__m</code>	The mask to compare against.

ReturnsTrue if `__m & table[__c]` is true, false otherwise.**narrow()** [1/2]

```
char std::ctype< char >::narrow (
    char_type __c,
    char __dfault ) const [inline]
```

Narrow char.

This function converts the char to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. For an underived ctype<char> facet, c will be returned unchanged.

This function works as if it returns ctype<char>::do_narrow(c). do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted character.

References [std::ctype< _CharT >::do_narrow\(\)](#).**narrow()** [2/2]

```
const char_type * std::ctype< char >::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __dfault,
    char * __to ) const [inline]
```

Narrow char array.

This function converts each char in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *dfault* is used instead. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_narrow(lo, hi, dfault, to). do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::ctype<_CharT>::do_narrow\(\)](#).

scan_is()

```
const char * std::ctype< char >::scan_is (
    mask __m,
    const char * __lo,
    const char * __hi ) const [inline]
```

Find char matching a mask.

This function searches for and returns the first char in [lo,hi) for which is(m,char) is true.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char if found, else `__hi`.




scan_not()

```
const char * std::ctype< char >::scan_not (
    mask __m,
    const char * __lo,
    const char * __hi ) const [inline]
```

Find char not matching a mask.

This function searches for and returns a pointer to the first char in [__lo,__hi) for which is(m,char) is false.

Parameters

 _m	The mask to compare against.
 _lo	Pointer to start of range.
 _hi	Pointer to end of range.

Returns

Pointer to a non-matching char if found, else __hi.

table()

```
const mask * std::ctype< char >::table ( ) const throw ( ) [inline]
```

Returns a pointer to the mask table provided to the constructor, or the default from classic_table() if none was provided.

tolower() [1/2]



```
const char_type * std::ctype< char >::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline]
```

Convert array to lowercase.

This function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

tolower() acts as if it returns ctype<char>::do_tolower(__lo, __hi). do_tolower() must always return the same result for the same input.

Parameters

 _lo	Pointer to first char in range.
 _hi	Pointer to end of range.

Returns

__hi.

References [std::ctype< _CharT >::do_tolower\(\)](#).

tolower() [2/2]


```
char_type std::ctype< char >::tolower (
    char_type __c ) const [inline]
```

Convert to lowercase.

This function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

tolower() acts as if it returns ctype<char>::do_tolower(__c). do_tolower() must always return the same result for the same input.

Parameters

 _c	The char to convert.
---	----------------------

Returns

The lowercase char if convertible, else `__c`.

References [std::ctype<_CharT>::do_tolower\(\)](#).

toupper() [1/2]

```
const char_type * std::ctype< char >::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline]
```

Convert array to uppercase.

This function converts each char in the range `[__lo,__hi)` to uppercase if possible. Other chars remain untouched.

`toupper()` acts as if it returns `ctype<char>::do_toupper(__lo, __hi)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

References [std::ctype<_CharT>::do_toupper\(\)](#).

toupper() [2/2]

```
char_type std::ctype< char >::toupper (
    char_type __c ) const [inline]
```

Convert to uppercase.

This function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`toupper()` acts as if it returns `ctype<char>::do_toupper(c)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

References [std::ctype<_CharT>::do_toupper\(\)](#).

widen() [1/2]

```
char_type std::ctype< char >::widen (
    char __c ) const [inline]
```

Widen char.

This function converts the char to `char_type` using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be returned unchanged.

This function works as if it returns `ctype<char>::do_widen(c)`. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>_↔</code>	The char to convert.
<code>_c</code>	

Returns

The converted character.

References [std::ctype< _CharT >::do_widen\(\)](#).

widen() [2/2]

```
const char * std::ctype< char >::widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline]
```

Widen char array.

This function converts each char in the input to char using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be copied unchanged.

This function works as if it returns `ctype<char>::do_widen(c)`. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>_↔</code> <code>_lo</code>	Pointer to first char in range.
<code>_↔</code> <code>_hi</code>	Pointer to end of range.
<code>_↔</code> <code>_to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::ctype< _CharT >::do_widen\(\)](#).

6.380.5 Member Data Documentation

id

```
locale::id std::ctype< char >::id [static]
```

The facet id for `ctype<char>`

table_size

```
const size_t std::ctype< char >::table_size [static]
```

The size of the mask table. It is `SCHAR_MAX + 1`.

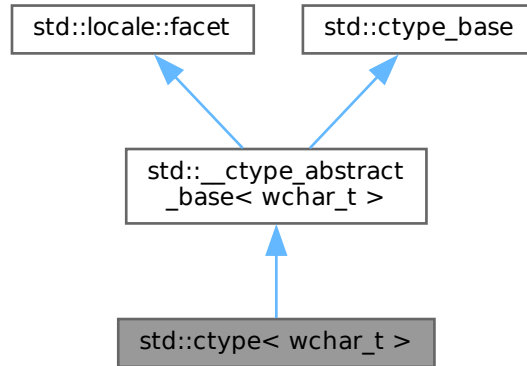
The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [ctype_inline.h](#)

6.381 std::ctype< wchar_t > Class Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::ctype< wchar_t >:



Public Types

- typedef const int * **__to_type**
- typedef wctype_t **__wmask_type**
- typedef wchar_t **char_type**
- typedef unsigned short **mask**

Public Member Functions

- [ctype](#) (__c_locale __cloc, size_t __refs=0)
- [ctype](#) (size_t __refs=0)
- const [char_type](#) * [is](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, mask * __vec) const
- bool [is](#) (mask __m, [char_type](#) __c) const
- char [narrow](#) ([char_type](#) __c, char __dfault) const
- const [char_type](#) * [narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, char __dfault, char * __to) const
- const [char_type](#) * [scan_is](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- const [char_type](#) * [scan_not](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- const [char_type](#) * [tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [tolower](#) ([char_type](#) __c) const
- const [char_type](#) * [toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [toupper](#) ([char_type](#) __c) const
- [char_type](#) [widen](#) (char __c) const
- const char * [widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual [~ctype](#) ()
- [__wmask_type](#) **_M_convert_to_wmask** (const mask __m) const throw ()
- void **_M_initialize_ctype** () throw ()
- virtual const [char_type](#) * **do_is** (const [char_type](#) *__lo, const [char_type](#) *__hi, mask *__vec) const
- virtual bool **do_is** (mask __m, [char_type](#) __c) const
- virtual [char](#) **do_narrow** ([char_type](#) __c, [char](#) __dfault) const
- virtual const [char_type](#) * **do_narrow** (const [char_type](#) *__lo, const [char_type](#) *__hi, [char](#) __dfault, [char](#) *__to) const
- virtual const [char_type](#) * **do_scan_is** (mask __m, const [char_type](#) *__lo, const [char_type](#) *__hi) const
- virtual const [char_type](#) * **do_scan_not** (mask __m, const [char_type](#) *__lo, const [char_type](#) *__hi) const
- virtual const [char_type](#) * **do_tolower** ([char_type](#) *__lo, const [char_type](#) *__hi) const
- virtual [char_type](#) **do_tolower** ([char_type](#) __c) const
- virtual const [char_type](#) * **do_toupper** ([char_type](#) *__lo, const [char_type](#) *__hi) const
- virtual [char_type](#) **do_toupper** ([char_type](#) __c) const
- virtual [char_type](#) **do_widen** ([char](#) __c) const
- virtual const [char](#) * **do_widen** (const [char](#) *__lo, const [char](#) *__hi, [char_type](#) *__to) const

Static Protected Member Functions

- static [__c_locale](#) **_S_clone_c_locale** ([__c_locale](#) &__cloc) throw ()
- static void **_S_create_c_locale** ([__c_locale](#) &__cloc, const [char](#) *__s, [__c_locale](#) __old=0)
- static void **_S_destroy_c_locale** ([__c_locale](#) &__cloc)
- static [__c_locale](#) **_S_get_c_locale** ()
- static const [char](#) * **_S_get_c_name** () throw ()
- static [__c_locale](#) **_S_lc_ctype_c_locale** ([__c_locale](#) __cloc, const [char](#) *__s)

Protected Attributes

- mask **_M_bit** [16]
- [__c_locale](#) **_M_c_locale_ctype**
- [char](#) **_M_narrow** [128]
- bool **_M_narrow_ok**
- [wint_t](#) **_M_widen** [1+static_cast< unsigned char >(-1)]
- [__wmask_type](#) **_M_wmask** [16]

6.381.1 Detailed Description

The `ctype<wchar_t>` specialization.

This class defines classification and conversion functions for the `wchar_t` type. It gets used by `wchar_t` streams for many I/O operations. The `wchar_t` specialization provides a number of optimizations as well. `ctype<wchar_t>` inherits its public methods from `__ctype_abstract_base<wchar_t>`.

6.381.2 Member Typedef Documentation

char_type

```
typedef wchar_t std::ctype< wchar_t >::char_type
```

Typedef for the template parameter `wchar_t`.

6.381.3 Constructor & Destructor Documentation

ctype() [1/2]

```
std::ctype< wchar_t >::ctype (
    size_t __refs = 0 ) [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

ctype() [2/2]

```
std::ctype< wchar_t >::ctype (
    __c_locale __cloc,
    size_t __refs = 0 ) [explicit]
```

Constructor performs static initialization.

This constructor is used to construct the initial C locale facet.

Parameters

<code>__cloc</code>	Handle to C locale data.
<code>__refs</code>	Passed to the base facet class.

~ctype()

```
virtual std::ctype< wchar_t >::~~ctype ( ) [protected], [virtual]
```

Destructor.

6.381.4 Member Function Documentation

do_is() [1/2]

```
virtual const char_type * std::ctype< wchar_t >::do_is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [protected], [virtual]
```

Return a mask array.

This function finds the mask for each `wchar_t` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the input.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_is() [2/2]

```
virtual bool std::ctype< wchar_t >::do_is (
    mask __m,
    char_type __c ) const [protected], [virtual]
```

Test `wchar_t` classification.

This function finds a mask `M` for `c` and compares it to mask `m`.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>wchar_t</code> to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_narrow() [1/2]

```
virtual char std::ctype< wchar_t >::do_narrow (
    char_type __c,
    char __dfault ) const [protected], [virtual]
```

Narrow `wchar_t` to `char`.

This virtual function converts the argument to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an underived `ctype<wchar_t>` facet, `c` will be cast to `char` and returned.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>wchar_t</code> to convert.
------------------	--------------------------------------

Parameters

<code>__default</code>	Char to return if conversion fails.
------------------------	-------------------------------------

Returns

The converted char.

Implements `std::__ctype_abstract_base< wchar_t >`.

do_narrow() [2/2]

```
virtual const char_type * std::__ctype< wchar_t >::do_narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [protected], [virtual]
```

Narrow `wchar_t` array to char array.

This virtual function converts each `wchar_t` in the range `[lo,hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any `wchar_t` in the input that cannot be converted, *default* is used instead. For an undervied `ctype<wchar_t>` facet, the argument will be copied, casting each element to `char`.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements `std::__ctype_abstract_base< wchar_t >`.

do_scan_is()

```
virtual const char_type * std::__ctype< wchar_t >::do_scan_is (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Find `wchar_t` matching mask.

This function searches for and returns the first `wchar_t` `c` in `[__lo,__hi)` for which `is(__m,c)` is true.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching `wchar_t` if found, else `__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_scan_not()

```
virtual const char_type * std::ctype< wchar_t >::do_scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Find `wchar_t` not matching mask.

This function searches for and returns a pointer to the first `wchar_t` `c` of `[__lo,__hi)` for which `is(__m,c)` is false.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching `wchar_t` if found, else `__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_tolower() [1/2]

```
virtual const char_type * std::ctype< wchar_t >::do_tolower (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual]
```

Convert array to lowercase.

This virtual function converts each `wchar_t` in the range `[lo,hi)` to lowercase if possible. Other elements remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_tolower() [2/2]

```
virtual char_type std::ctype< wchar_t >::do_tolower (
    char_type __c ) const    [protected], [virtual]
```

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

<code>__c</code>	The wchar_t to convert.
------------------	-------------------------

Returns

The lowercase wchar_t if convertible, else `__c`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_toupper() [1/2]

```
virtual const char_type * std::ctype< wchar_t >::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const    [protected], [virtual]
```

Convert array to uppercase.

This virtual function converts each wchar_t in the range [lo,hi) to uppercase if possible. Other elements remain untouched.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

do_toupper() [2/2]

```
virtual char_type std::ctype< wchar_t >::do_toupper (
    char_type __c ) const    [protected], [virtual]
```

Convert to uppercase.

This virtual function converts the `wchar_t` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>wchar_t</code> to convert.
------------------	--------------------------------------

Returns

The uppercase `wchar_t` if convertible, else `__c`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

`do_widen()` [1/2]

```
virtual char_type std::ctype< wchar_t >::do_widen (
    char __c ) const [protected], [virtual]
```

Widen `char` to `wchar_t`.

This virtual function converts the `char` to `wchar_t` using the simplest reasonable transformation. For an underived `ctype<wchar_t>` facet, the argument will be cast to `wchar_t`.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>char</code> to convert.
------------------	-----------------------------------

Returns

The converted `wchar_t`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

`do_widen()` [2/2]

```
virtual const char * std::ctype< wchar_t >::do_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [protected], [virtual]
```

Widen `char` array to `wchar_t` array.

This function converts each `char` in the input to `wchar_t` using the simplest reasonable transformation. For an underived `ctype<wchar_t>` facet, the argument will be copied, casting each element to `wchar_t`.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start range.
-------------------	-------------------------

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

[`__hi`](#).

Implements [std::__ctype_abstract_base< wchar_t >](#).

is() [1/2]

```
const char_type * std::__ctype_abstract_base< wchar_t >::is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [inline], [inherited]
```

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the char array. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

[`__hi`](#).

is() [2/2]

```
bool std::__ctype_abstract_base< wchar_t >::is (
    mask __m,
    char_type __c ) const [inline], [inherited]
```

Test char_type classification.

This function finds a mask M for __c and compares it to mask __m. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__c</code>	The char_type to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

(M & __m) != 0.

narrow() [1/2]

```
char std::__ctype_abstract_base< wchar_t >::narrow (
    char_type __c,
    char __default ) const [inline], [inherited]
```

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, default is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

__c	The char_type to convert.
__default	Char to return if conversion fails.

Returns

The converted char.

narrow() [2/2]

```
const char_type * std::__ctype_abstract_base< wchar_t >::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline], [inherited]
```

Narrow array to char array.

This function converts each char_type in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char_type in the input that cannot be converted, default is used instead. It does so by returning ctype<char_type>::do_narrow(__lo, __hi, __default, __to).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

__lo	Pointer to start of range.
__hi	Pointer to end of range.
__default	Char to use if conversion fails.
__to	Pointer to the destination array.

Returns

__hi.

scan_is()

```
const char_type * std::__ctype_abstract_base< wchar_t >::scan_is (
    mask __m,
```

```
const char_type * __lo,
const char_type * __hi ) const [inline], [inherited]
```

Find char_type matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is true. It does so by returning ctype<char_type>::do_scan_is().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else `__hi`.

scan_not()

```
const char_type * std::__ctype_abstract_base< wchar_t >::scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to non-matching char if found, else `__hi`.

tolower() [1/2]

```
const char_type * std::__ctype_abstract_base< wchar_t >::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to lowercase.

This function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_tolower(__lo,__hi).

Parameters

$\begin{smallmatrix} \leftarrow \\ _lo \end{smallmatrix}$	Pointer to start of range.
$\begin{smallmatrix} \leftarrow \\ _hi \end{smallmatrix}$	Pointer to end of range.

Returns

$_hi$.

tolower() [2/2]

```
char_type std::__ctype_abstract_base< wchar_t >::tolower (
    char_type __c ) const [inline], [inherited]
```

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_tolower(c).

Parameters

$\begin{smallmatrix} \leftarrow \\ _c \end{smallmatrix}$	The char_type to convert.
---	---------------------------

Returns

The lowercase char_type if convertible, else $_c$.

toupper() [1/2]

```
const char_type * std::__ctype_abstract_base< wchar_t >::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

$\begin{smallmatrix} \leftarrow \\ _lo \end{smallmatrix}$	Pointer to start of range.
$\begin{smallmatrix} \leftarrow \\ _hi \end{smallmatrix}$	Pointer to end of range.

Returns

$_hi$.

toupper() [2/2]

```
char_type std::__ctype_abstract_base< wchar_t >::toupper (
    char_type __c ) const [inline], [inherited]
```

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_toupper()`.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The uppercase `char_type` if convertible, else `__c`.

widen() [1/2]

```
char_type std::__ctype_abstract_base< wchar_t >::widen (  
    char __c ) const    [inline], [inherited]
```

Widen char to `char_type`.

This function converts the char argument to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted `char_type`.

widen() [2/2]

```
const char * std::__ctype_abstract_base< wchar_t >::widen (  
    const char * __lo,  
    const char * __hi,  
    char_type * __to ) const    [inline], [inherited]
```

Widen array to `char_type`.

This function converts each char in the input to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

6.381.5 Member Data Documentation

id

```
locale::id std::ctype< wchar_t >::id [static]
```

The facet id for `ctype<wchar_t>`

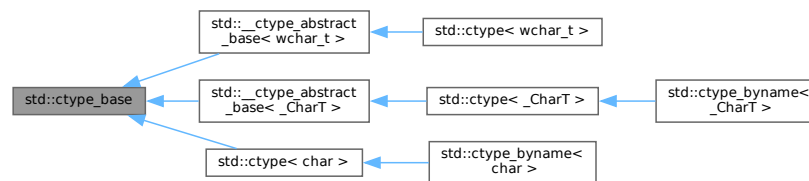
The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.382 std::ctype_base Struct Reference

```
#include <ctype_base.h>
```

Inheritance diagram for `std::ctype_base`:



Public Types

- typedef const int * **__to_type**
- typedef unsigned short **mask**

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

6.382.1 Detailed Description

Base class for `ctype`.

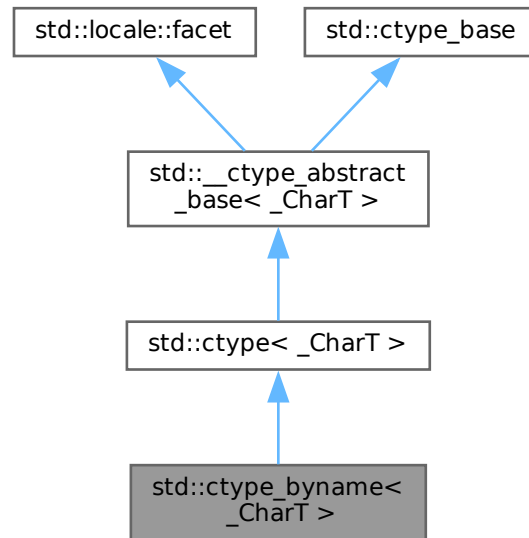
The documentation for this struct was generated from the following file:

- [ctype_base.h](#)

6.383 std::ctype_byname<_CharT> Class Template Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::ctype_byname<_CharT>:



Public Types

- typedef const int * **__to_type**
- typedef _CharT **char_type**
- typedef `ctype<_CharT>::mask` **mask**

Public Member Functions

- **ctype_byname** (const char *__s, size_t __refs=0)
- **ctype_byname** (const [string](#) &__s, size_t __refs=0)
- const char_type * **is** (const char_type *__lo, const char_type *__hi, mask *__vec) const
- bool **is** (mask __m, char_type __c) const
- char **narrow** (char_type __c, char __dfault) const
- const char_type * **narrow** (const char_type *__lo, const char_type *__hi, char __dfault, char *__to) const
- const char_type * **scan_is** (mask __m, const char_type *__lo, const char_type *__hi) const
- const char_type * **scan_not** (mask __m, const char_type *__lo, const char_type *__hi) const
- const char_type * **tolower** (char_type *__lo, const char_type *__hi) const
- char_type **tolower** (char_type __c) const
- const char_type * **toupper** (char_type *__lo, const char_type *__hi) const
- char_type **toupper** (char_type __c) const
- char_type **widen** (char __c) const
- const char * **widen** (const char *__lo, const char *__hi, char_type *__to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual const char_type * [do_is](#) (const char_type * __lo, const char_type * __hi, mask * __vec) const
- virtual bool [do_is](#) (mask __m, char_type __c) const
- virtual char [do_narrow](#) (char_type, char __dfault) const
- virtual const char_type * [do_narrow](#) (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const
- virtual const char_type * [do_scan_is](#) (mask __m, const char_type * __lo, const char_type * __hi) const
- virtual const char_type * [do_scan_not](#) (mask __m, const char_type * __lo, const char_type * __hi) const
- virtual const char_type * [do_tolower](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_tolower](#) (char_type __c) const
- virtual const char_type * [do_toupper](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_toupper](#) (char_type __c) const
- virtual char_type [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, char_type * __dest) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char * __s)

6.383.1 Detailed Description

```
template<typename _CharT>
class std::ctype_byname<_CharT>
```

class ctype_byname [22.2.1.2].

6.383.2 Member Function Documentation

do_is() [1/2]

```
template<typename _CharT >
virtual const char_type * std::ctype< _CharT >::do_is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [protected], [virtual], [inherited]
```

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the input.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< _CharT >](#).

do_is() [2/2]

```
template<typename _CharT >
virtual bool std::ctype< _CharT >::do_is (
    mask __m,
    char_type __c ) const [protected], [virtual], [inherited]
```

Test char_type classification.

This function finds a mask M for c and compares it to mask m.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__c</code>	The char_type to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

$(M \& _m) \neq 0$.

Implements [std::__ctype_abstract_base< _CharT >](#).

do_narrow() [1/2]

```
template<typename _CharT >
virtual char std::ctype< _CharT >::do_narrow (
```

```
char_type __c,
char __default ) const [protected], [virtual], [inherited]
```

Narrow char_type to char.

This virtual function converts the argument to char using the simplest reasonable transformation. If the conversion fails, `__default` is returned instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::narrow\(\)](#), and [std::ctype<char>::narrow\(\)](#).

do_narrow() [2/2]

```
template<typename _CharT>
virtual const char_type * std::ctype<_CharT>::do_narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [protected], [virtual], [inherited]
```

Narrow char_type array to char.

This virtual function converts each char_type in the range `[__lo,__hi)` to char using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__default` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_scan_is()

```
template<typename _CharT>
virtual const char_type * std::ctype<_CharT>::do_scan_is (
```

```

mask __m,
const char_type * __lo,
const char_type * __hi ) const [protected], [virtual], [inherited]

```

Find char_type matching mask.

This function searches for and returns the first char_type c in [__lo,__hi) for which is(__m,c) is true.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char_type if found, else __hi.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_scan_not()

```

template<typename _CharT >
virtual const char_type * std::__ctype<_CharT>::do_scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [protected], [virtual], [inherited]

```

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char_type if found, else __hi.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_tolower() [1/2]

```

template<typename _CharT >
virtual const char_type * std::__ctype<_CharT>::do_tolower (

```



```
char_type * __lo,
const char_type * __hi ) const [protected], [virtual], [inherited]
```

Convert array to lowercase.

This virtual function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

 <code>__lo</code>	Pointer to start of range.
 <code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::ctype_abstract_base<_CharT>](#).

do_tolower() [2/2]


```
template<typename _CharT>
virtual char_type std::ctype<_CharT>::do_tolower (
    char_type __c ) const [protected], [virtual], [inherited]
```

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

 <code>__c</code>	The char_type to convert.
--	---------------------------

Returns

The lowercase char_type if convertible, else __c.

Implements [std::ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::tolower\(\)](#), and [std::ctype<char>::tolower\(\)](#).

do_toupper() [1/2]



```
template<typename _CharT>
virtual const char_type * std::ctype<_CharT>::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual], [inherited]
```

Convert array to uppercase.

This virtual function converts each char_type in the range [__lo,__hi) to uppercase if possible. Other elements remain untouched.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

 <code>__lo</code>	Pointer to start of range.
 <code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

do_toupper() [2/2]


```
template<typename _CharT >
virtual char_type std::ctype<_CharT>::do_toupper (
    char_type __c ) const [protected], [virtual], [inherited]
```

Convert to uppercase.

This virtual function converts the `char_type` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

 <code>__c</code>	The <code>char_type</code> to convert.
---	--

Returns

The uppercase `char_type` if convertible, else `__c`.

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::toupper\(\)](#), and [std::ctype<char>::toupper\(\)](#).

do_widen() [1/2]

```
template<typename _CharT >
virtual char_type std::ctype<_CharT>::do_widen (
    char __c ) const [protected], [virtual], [inherited]
```


Widen char.

This virtual function converts the `char` to `char_type` using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

 <code>__c</code>	The <code>char</code> to convert.
---	-----------------------------------

Returns

The converted char_type

Implements [std::__ctype_abstract_base<_CharT>](#).

Referenced by [std::ctype<char>::widen\(\)](#), and [std::ctype<char>::widen\(\)](#).

do_widen() [2/2]

```
template<typename _CharT >
virtual const char * std::ctype<\_CharT>::do\_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [protected], [virtual], [inherited]
```

Widen char array.

This function converts each char in the input to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

__lo	Pointer to start range.
__hi	Pointer to end of range.
__to	Pointer to the destination array.

Returns

[__hi](#).

Implements [std::__ctype_abstract_base<_CharT>](#).

is() [1/2]

```
template<typename _CharT >
const char_type * std::\_\_ctype\_abstract\_base<\_CharT>::is (
    const char_type * __lo,
    const char_type * __hi,
    mask * __vec ) const [inline], [inherited]
```

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the char array. It does so by returning the value of ctype<char_type>::do_is().

Parameters

__lo	Pointer to start of range.
__hi	Pointer to end of range.
__vec	Pointer to an array of mask storage.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_is\(\)](#).

is() [2/2]

```
template<typename _CharT >
bool std::__ctype_abstract_base<_CharT>::is (
    mask __m,
    char_type __c ) const [inline], [inherited]
```

Test char_type classification.

This function finds a mask M for `__c` and compares it to mask `__m`. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

<code>__c</code>	The char_type to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0`.

References [std::__ctype_abstract_base<_CharT>::do_is\(\)](#).

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), and [std::time_get<_CharT, _InIter>::get\(\)](#).

narrow() [1/2]

```
template<typename _CharT >
char std::__ctype_abstract_base<_CharT>::narrow (
    char_type __c,
    char __dfault ) const [inline], [inherited]
```

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. It does so by returning `ctype<char_type>::do_narrow(__c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::do_get_year\(\)](#), [std::time_get<_CharT, _InIter>::get\(\)](#), and [std::time_put<_CharT, _OutIter>::put\(\)](#).

narrow() [2/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline], [inherited]
```

Narrow array to char array.

This function converts each char_type in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char_type in the input that cannot be converted, *default* is used instead. It does so by returning ctype<char_type>::do_narrow(__lo, __hi, __default, __to).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_narrow\(\)](#).

scan_is()

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT>::scan_is (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Find char_type matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is true. It does so by returning ctype<char_type>::do_scan_is().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_is\(\)](#).




scan_not()

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT >::scan_not (
    mask __m,
    const char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

 __m	The mask to compare against.
 __lo	Pointer to first char in range.
 __hi	Pointer to end of range.

Returns

Pointer to non-matching char if found, else __hi.

References [std::__ctype_abstract_base<_CharT>::do_scan_not\(\)](#).



tolower() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT >::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to lowercase.

This function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>:: do_tolower(__lo, __hi).

Parameters

 __lo	Pointer to start of range.
 __hi	Pointer to end of range.

Returns

__hi.

References [std::__ctype_abstract_base<_CharT>::do_tolower\(\)](#).

tolower() [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT >::tolower (
    char_type __c ) const [inline], [inherited]
```

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_tolower(c).

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else __c.

References [std::__ctype_abstract_base<_CharT>::do_tolower\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get\(\)](#).

toupper() [1/2]

```
template<typename _CharT >
const char_type * std::__ctype_abstract_base<_CharT >::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

References [std::__ctype_abstract_base<_CharT>::do_toupper\(\)](#).

toupper() [2/2]

```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT >::toupper (
    char_type __c ) const [inline], [inherited]
```

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_toupper(c).

Parameters

  <i>c</i>	The char_type to convert.
---	---------------------------

Returns

The uppercase char_type if convertible, else __c.

References [std::__ctype_abstract_base<_CharT>::do_toupper\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get\(\)](#).

widen() [1/2]



```
template<typename _CharT >
char_type std::__ctype_abstract_base<_CharT>::widen (
    char __c ) const [inline], [inherited]
```

Widen char to char_type.

This function converts the char argument to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

  <i>c</i>	The char to convert.
---	----------------------

Returns

The converted char_type.

References [std::__ctype_abstract_base<_CharT>::do_widen\(\)](#).

Referenced by [std::money_get<_CharT, _InIter>::do_get\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::money_put<_CharT, _OutIter>::do_put\(\)](#), and [std::time_put<_CharT, _OutIter>::do_put\(\)](#).

widen() [2/2]







```
template<typename _CharT >
const char * std::__ctype_abstract_base<_CharT>::widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline], [inherited]
```

Widen array to char_type.

This function converts each char in the input to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

  <i>lo</i>	Pointer to start of range.
  <i>hi</i>	Pointer to end of range.
  <i>to</i>	Pointer to the destination array.

Returns

`__hi`.

References [std::__ctype_abstract_base< _CharT >::do_widen\(\)](#).

6.383.3 Member Data Documentation

id

```
template<typename _CharT >
locale::id std::ctype< _CharT >::id [static], [inherited]
The facet id for ctype<char_type>
```

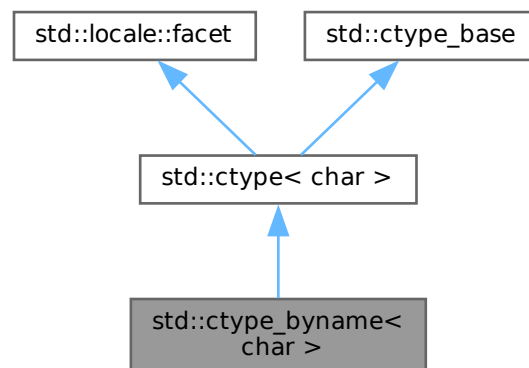
The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.384 std::ctype_byname< char > Class Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::ctype_byname< char >:



Public Types

- typedef const int * `__to_type`
- typedef char `char_type`
- typedef unsigned short `mask`

Public Member Functions

- `ctype_byname` (const char * __s, size_t __refs=0)
- `ctype_byname` (const [string](#) & __s, size_t __refs=0)
- const char * `is` (const char * __lo, const char * __hi, mask * __vec) const
- bool `is` (mask __m, char __c) const
- char `narrow` (char_type __c, char __default) const
- const `char_type` * `narrow` (const `char_type` * __lo, const `char_type` * __hi, char __default, char * __to) const

- const char * [scan_is](#) (mask __m, const char * __lo, const char * __hi) const
- const char * [scan_not](#) (mask __m, const char * __lo, const char * __hi) const
- const mask * [table](#) () const throw ()
- const [char_type](#) * [tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [tolower](#) ([char_type](#) __c) const
- const [char_type](#) * [toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [toupper](#) ([char_type](#) __c) const
- [char_type](#) [widen](#) (char __c) const
- const char * [widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Public Member Functions

- static const mask * [classic_table](#) () throw ()

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const size_t [table_size](#)
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual char [do_narrow](#) ([char_type](#) __c, char __dfault) const
- virtual const [char_type](#) * [do_narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, char __dfault, char * __to) const
- virtual const [char_type](#) * [do_tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_tolower](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_toupper](#) ([char_type](#) __c) const
- virtual [char_type](#) [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char * __s)

Protected Attributes

- `__c_locale` `_M_c_locale_ctype`
- `bool` `_M_del`
- `char` `_M_narrow` [1+static_cast< unsigned char >(-1)]
- `char` `_M_narrow_ok`
- `const mask *` `_M_table`
- `__to_type` `_M_tolower`
- `__to_type` `_M_toupper`
- `char` `_M_widen` [1+static_cast< unsigned char >(-1)]
- `char` `_M_widen_ok`

6.384.1 Detailed Description

22.2.1.4 Class `ctype_byname` specializations.

6.384.2 Member Typedef Documentation**char_type**

typedef char `std::ctype< char >::char_type` [inherited]
 Typedef for the template parameter char.

6.384.3 Member Function Documentation**classic_table()**

static const mask * `std::ctype< char >::classic_table () throw ()` [static], [inherited]
 Returns a pointer to the C locale mask table.

do_narrow() [1/2]

```
virtual char std::ctype< char >::do_narrow (  

    char_type __c,  

    char __dfault ) const [inline], [protected], [virtual], [inherited]
```

Narrow char.

This virtual function converts the char to char using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an underived `ctype<char>` facet, `c` will be returned unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

do_narrow() [2/2]

```
virtual const char_type * std::ctype< char >::do_narrow (  

    const char_type * __lo,
```



```
const char_type * __hi,
char __default,
char * __to ) const [inline], [protected], [virtual], [inherited]
```

Narrow char array to char array.

This virtual function converts each char in the range [lo,hi) to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an undervived `ctype<char>` facet, the argument will be copied unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

`do_tolower()` [1/2]

```
virtual const char_type * std::ctype< char >::do_tolower (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual], [inherited]
```

Convert array to lowercase.

This virtual function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

`do_tolower()` [2/2]

```
virtual char_type std::ctype< char >::do_tolower (
    char_type __c ) const [protected], [virtual], [inherited]
```

Convert to lowercase.

This virtual function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The lowercase char if convertible, else `__c`.

do_toupper() [1/2]

```
virtual const char_type * std::ctype< char >::do_toupper (
    char_type * __lo,
    const char_type * __hi ) const [protected], [virtual], [inherited]
```

Convert array to uppercase.

This virtual function converts each char in the range [lo,hi) to uppercase if possible. Other chars remain untouched.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

do_toupper() [2/2]

```
virtual char_type std::ctype< char >::do_toupper (
    char_type __c ) const [protected], [virtual], [inherited]
```

Convert to uppercase.

This virtual function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

do_widen() [1/2]

```
virtual char_type std::ctype< char >::do_widen (
```

```
char __c ) const [inline], [protected], [virtual], [inherited]
```

Widen char.

This virtual function converts the char to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be returned unchanged.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

\leftrightarrow __c	The char to convert.
--------------------------	----------------------

Returns

The converted character.

do_widen() [2/2]

```
virtual const char * std::ctype< char >::do_widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline], [protected], [virtual], [inherited]
```

Widen char array.

This function converts each char in the range [lo,hi) to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be copied unchanged.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

\leftrightarrow __lo	Pointer to start of range.
\leftrightarrow __hi	Pointer to end of range.
\leftrightarrow __to	Pointer to the destination array.

Returns

__hi.

is() [1/2]

```
const char * std::ctype< char >::is (
    const char * __lo,
    const char * __hi,
    mask * __vec ) const [inline], [inherited]
```

Return a mask array.

This function finds the mask for each char in the range [lo, hi) and successively writes it to vec. vec must have as many elements as the char array.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

is() [2/2]

```
bool std::ctype< char >::is (
    mask __m,
    char __c ) const [inline], [inherited]
```

Test char classification.

This function compares the mask table[c] to `__m`.

Parameters

<code>__c</code>	The char to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

True if `__m & table[__c]` is true, false otherwise.

narrow() [1/2]

```
char std::ctype< char >::narrow (
    char_type __c,
    char __dfault ) const [inline], [inherited]
```

Narrow char.

This function converts the char to char using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an underived `ctype<char>` facet, `c` will be returned unchanged.

This function works as if it returns `ctype<char>::do_narrow(c)`. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted character.

References [std::ctype< _CharT >::do_narrow\(\)](#).

narrow() [2/2]

```
const char_type * std::ctype< char >::narrow (
    const char_type * __lo,
    const char_type * __hi,
    char __default,
    char * __to ) const [inline], [inherited]
```

Narrow char array.

This function converts each char in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an underived `ctype<char>` facet, the argument will be copied unchanged.

This function works as if it returns `ctype<char>::do_narrow(lo, hi, default, to)`. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecv` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References `std::ctype< _CharT >::do_narrow()`.

scan_is()

```
const char * std::ctype< char >::scan_is (
    mask __m,
    const char * __lo,
    const char * __hi ) const [inline], [inherited]
```

Find char matching a mask.

This function searches for and returns the first char in `[lo,hi)` for which `is(m,char)` is true.

Parameters

<code>↔ __m</code>	The mask to compare against.
<code>↔ __lo</code>	Pointer to start of range.
<code>↔ __hi</code>	Pointer to end of range.

Returns

Pointer to a matching char if found, else `__hi`.

scan_not()

```
const char * std::ctype< char >::scan_not (
    mask __m,
```

```
const char * __lo,
const char * __hi ) const [inline], [inherited]
```

Find char not matching a mask.

This function searches for and returns a pointer to the first char in [__lo,__hi) for which is(m,char) is false.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char if found, else __hi.

table()

```
const mask * std::ctype< char >::table ( ) const throw ( ) [inline], [inherited]
```

Returns a pointer to the mask table provided to the constructor, or the default from classic_table() if none was provided.

tolower() [1/2]

```
const char_type * std::ctype< char >::tolower (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to lowercase.

This function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

tolower() acts as if it returns ctype<char>::do_tolower(__lo, __hi). do_tolower() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

__hi.

References [std::ctype< _CharT >::do_tolower\(\)](#).

tolower() [2/2]

```
char_type std::ctype< char >::tolower (
    char_type __c ) const [inline], [inherited]
```

Convert to lowercase.

This function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

tolower() acts as if it returns ctype<char>::do_tolower(__c). do_tolower() must always return the same result for the same input.

Parameters

	The char to convert.
<code>__c</code>	

Returns

The lowercase char if convertible, else `__c`.

References [std::ctype<_CharT>::do_tolower\(\)](#).

toupper() [1/2]



```
const char_type * std::ctype< char >::toupper (
    char_type * __lo,
    const char_type * __hi ) const [inline], [inherited]
```

Convert array to uppercase.

This function converts each char in the range `[__lo,__hi)` to uppercase if possible. Other chars remain untouched.

`toupper()` acts as if it returns `ctype<char>::do_toupper(__lo, __hi)`. `do_toupper()` must always return the same result for the same input.

Parameters

	Pointer to first char in range.
<code>__lo</code>	
	Pointer to end of range.
<code>__hi</code>	

Returns

`__hi`.

References [std::ctype<_CharT>::do_toupper\(\)](#).

toupper() [2/2]


```
char_type std::ctype< char >::toupper (
    char_type __c ) const [inline], [inherited]
```

Convert to uppercase.

This function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`toupper()` acts as if it returns `ctype<char>::do_toupper(c)`. `do_toupper()` must always return the same result for the same input.

Parameters

	The char to convert.
<code>__c</code>	

Returns

The uppercase char if convertible, else `__c`.

References [std::ctype< _CharT >::do_toupper\(\)](#).

widen() [1/2]

```
char_type std::ctype< char >::widen (
    char __c ) const [inline], [inherited]
```

Widen char.

This function converts the char to char_type using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be returned unchanged.

This function works as if it returns ctype<char>::do_widen(c). do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

References [std::ctype< _CharT >::do_widen\(\)](#).

widen() [2/2]

```
const char * std::ctype< char >::widen (
    const char * __lo,
    const char * __hi,
    char_type * __to ) const [inline], [inherited]
```

Widen char array.

This function converts each char in the input to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_widen(c). do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

References [std::ctype< _CharT >::do_widen\(\)](#).

6.384.4 Member Data Documentation

id

`locale::id std::ctype< char >::id` [static], [inherited]

The facet id for ctype<char>

table_size

`const size_t std::ctype< char >::table_size` [static], [inherited]

The size of the mask table. It is SCHAR_MAX + 1.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.385 std::ranges::dangling Struct Reference

```
#include <ranges_base.h>
```

Public Member Functions

- `template<typename... _Args>`
`constexpr dangling (_Args &&...) noexcept`

6.385.1 Detailed Description

Type returned by algorithms instead of a dangling iterator or subrange.

The documentation for this struct was generated from the following file:

- [ranges_base.h](#)

6.386 __gnu_cxx::debug_allocator< _Alloc > Class Template Reference

```
#include <debug_allocator.h>
```

Public Types

- `typedef _Traits::const_pointer const_pointer`
- `typedef _Traits::const_reference const_reference`
- `typedef _Traits::difference_type difference_type`
- `typedef _Traits::pointer pointer`
- `typedef _Traits::reference reference`
- `typedef _Traits::size_type size_type`
- `typedef _Traits::value_type value_type`

Public Member Functions

- `debug_allocator (const _Alloc &__a)`
- `template<typename _Alloc2 >`
`debug_allocator (const debug_allocator< _Alloc2 > &__a2, typename __convertible< _Alloc2 >::__type=0)`
- `pointer allocate (size_type __n)`
- `pointer allocate (size_type __n, const void * __hint)`
- `template<typename _Tp, typename... _Args>`
`void construct (_Tp * __p, _Args &&... __args)`
- `void construct (pointer __p, const value_type & __val)`
- `void deallocate (pointer __p, size_type __n)`

- `template<typename _Tp>`
`void destroy (_Tp *__p)`
- `size_type max_size () const throw ()`

Friends

- `template<typename _Alloc2>`
`bool operator== (const debug_allocator &__lhs, const debug_allocator<_Alloc2> &__rhs) noexcept`

6.386.1 Detailed Description

`template<typename _Alloc>`
`class __gnu_cxx::debug_allocator<_Alloc>`

A meta-allocator with debugging bits.

This is precisely the allocator defined in the C++03 Standard.

The documentation for this class was generated from the following file:

- [debug_allocator.h](#)

6.387 `std::decay<_Tp>` Class Template Reference

Public Types

- `typedef __decay_selector<__remove_type>::__type type`

6.387.1 Detailed Description

`template<typename _Tp>`
`class std::decay<_Tp>`

`decay`

The documentation for this class was generated from the following file:

- [type_traits](#)

6.388 `std::decimal::decimal128` Class Reference

Public Types

- `typedef float __decfloat128`

Public Member Functions

- [decimal128](#) (`__decfloat128 __z`)
- `decimal128 (decimal32 d32)`
- `decimal128 (decimal64 d64)`
- `decimal128 (double __r)`
- `decimal128 (float __r)`
- `decimal128 (int __z)`
- `decimal128 (long __z)`
- `decimal128 (long double __r)`
- `decimal128 (long long __z)`
- `decimal128 (unsigned int __z)`
- `decimal128 (unsigned long __z)`
- `decimal128 (unsigned long long __z)`

- `__decfloat128 __getval` (void)
- `void __setval` (`__decfloat128 __x`)
- **operator long long** () const
- `decimal128 & operator*=` (`decimal128 __rhs`)
- `decimal128 & operator*=` (`decimal32 __rhs`)
- `decimal128 & operator*=` (`decimal64 __rhs`)
- `decimal128 & operator*=` (int `__rhs`)
- `decimal128 & operator*=` (long `__rhs`)
- `decimal128 & operator*=` (long long `__rhs`)
- `decimal128 & operator*=` (unsigned int `__rhs`)
- `decimal128 & operator*=` (unsigned long `__rhs`)
- `decimal128 & operator*=` (unsigned long long `__rhs`)
- `decimal128 & operator++` ()
- `decimal128 operator++` (int)
- `decimal128 & operator+=` (`decimal128 __rhs`)
- `decimal128 & operator+=` (`decimal32 __rhs`)
- `decimal128 & operator+=` (`decimal64 __rhs`)
- `decimal128 & operator+=` (int `__rhs`)
- `decimal128 & operator+=` (long `__rhs`)
- `decimal128 & operator+=` (long long `__rhs`)
- `decimal128 & operator+=` (unsigned int `__rhs`)
- `decimal128 & operator+=` (unsigned long `__rhs`)
- `decimal128 & operator+=` (unsigned long long `__rhs`)
- `decimal128 & operator--` ()
- `decimal128 operator--` (int)
- `decimal128 & operator-=` (`decimal128 __rhs`)
- `decimal128 & operator-=` (`decimal32 __rhs`)
- `decimal128 & operator-=` (`decimal64 __rhs`)
- `decimal128 & operator-=` (int `__rhs`)
- `decimal128 & operator-=` (long `__rhs`)
- `decimal128 & operator-=` (long long `__rhs`)
- `decimal128 & operator-=` (unsigned int `__rhs`)
- `decimal128 & operator-=` (unsigned long `__rhs`)
- `decimal128 & operator-=` (unsigned long long `__rhs`)
- `decimal128 & operator/=` (`decimal128 __rhs`)
- `decimal128 & operator/=` (`decimal32 __rhs`)
- `decimal128 & operator/=` (`decimal64 __rhs`)
- `decimal128 & operator/=` (int `__rhs`)
- `decimal128 & operator/=` (long `__rhs`)
- `decimal128 & operator/=` (long long `__rhs`)
- `decimal128 & operator/=` (unsigned int `__rhs`)
- `decimal128 & operator/=` (unsigned long `__rhs`)
- `decimal128 & operator/=` (unsigned long long `__rhs`)

6.388.1 Detailed Description

3.2.4 Class decimal128.

6.388.2 Constructor & Destructor Documentation

decimal128()

```
std::decimal::decimal128::decimal128 (
    __decfloat128 __z ) [inline]
```

Conforming extension: Conversion from scalar decimal type.

The documentation for this class was generated from the following file:

- [decimal](#)

6.389 std::decimal::decimal32 Class Reference

Public Types

- typedef float **__decfloat32**

Public Member Functions

- [decimal32](#) (__decfloat32 __z)
- **decimal32** ([decimal128](#) __d128)
- **decimal32** ([decimal64](#) __d64)
- **decimal32** (double __r)
- **decimal32** (float __r)
- **decimal32** (int __z)
- **decimal32** (long __z)
- **decimal32** (long double __r)
- **decimal32** (long long __z)
- **decimal32** (unsigned int __z)
- **decimal32** (unsigned long __z)
- **decimal32** (unsigned long long __z)
- __decfloat32 **__getval** (void)
- void **__setval** (__decfloat32 __x)
- **operator long long** () const
- [decimal32](#) & **operator*=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator*=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator*=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator*=** (int __rhs)
- [decimal32](#) & **operator*=** (long __rhs)
- [decimal32](#) & **operator*=** (long long __rhs)
- [decimal32](#) & **operator*=** (unsigned int __rhs)
- [decimal32](#) & **operator*=** (unsigned long __rhs)
- [decimal32](#) & **operator*=** (unsigned long long __rhs)
- [decimal32](#) & **operator++** ()
- [decimal32](#) **operator++** (int)
- [decimal32](#) & **operator+=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator+=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator+=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator+=** (int __rhs)
- [decimal32](#) & **operator+=** (long __rhs)
- [decimal32](#) & **operator+=** (long long __rhs)
- [decimal32](#) & **operator+=** (unsigned int __rhs)
- [decimal32](#) & **operator+=** (unsigned long __rhs)

- [decimal32](#) & **operator+=** (unsigned long long __rhs)
- [decimal32](#) & **operator--** ()
- [decimal32](#) **operator--** (int)
- [decimal32](#) & **operator-=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator-=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator-=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator-=** (int __rhs)
- [decimal32](#) & **operator-=** (long __rhs)
- [decimal32](#) & **operator-=** (long long __rhs)
- [decimal32](#) & **operator-=** (unsigned int __rhs)
- [decimal32](#) & **operator-=** (unsigned long __rhs)
- [decimal32](#) & **operator-=** (unsigned long long __rhs)
- [decimal32](#) & **operator/=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator/=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator/=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator/=** (int __rhs)
- [decimal32](#) & **operator/=** (long __rhs)
- [decimal32](#) & **operator/=** (long long __rhs)
- [decimal32](#) & **operator/=** (unsigned int __rhs)
- [decimal32](#) & **operator/=** (unsigned long __rhs)
- [decimal32](#) & **operator/=** (unsigned long long __rhs)

6.389.1 Detailed Description

3.2.2 Class decimal32.

6.389.2 Constructor & Destructor Documentation

decimal32()

```
std::decimal::decimal32::decimal32 (
    __decfloat32 __z ) [inline]
```

Conforming extension: Conversion from scalar decimal type.

The documentation for this class was generated from the following file:

- [decimal](#)

6.390 std::decimal::decimal64 Class Reference

Public Types

- typedef float **__decfloat64**

Public Member Functions

- [decimal64](#) (__decfloat64 __z)
- **decimal64** ([decimal128](#) d128)
- **decimal64** ([decimal32](#) d32)
- **decimal64** (double __r)
- **decimal64** (float __r)
- **decimal64** (int __z)
- **decimal64** (long __z)
- **decimal64** (long double __r)
- **decimal64** (long long __z)

- **decimal64** (unsigned int __z)
- **decimal64** (unsigned long __z)
- **decimal64** (unsigned long long __z)
- **__decfloat64 __getval** (void)
- void **__setval** (__decfloat64 __x)
- **operator long long** () const
- **decimal64** & **operator*=** ([decimal128](#) __rhs)
- **decimal64** & **operator*=** ([decimal32](#) __rhs)
- **decimal64** & **operator*=** ([decimal64](#) __rhs)
- **decimal64** & **operator*=** (int __rhs)
- **decimal64** & **operator*=** (long __rhs)
- **decimal64** & **operator*=** (long long __rhs)
- **decimal64** & **operator*=** (unsigned int __rhs)
- **decimal64** & **operator*=** (unsigned long __rhs)
- **decimal64** & **operator*=** (unsigned long long __rhs)
- **decimal64** & **operator++** ()
- **decimal64** **operator++** (int)
- **decimal64** & **operator+=** ([decimal128](#) __rhs)
- **decimal64** & **operator+=** ([decimal32](#) __rhs)
- **decimal64** & **operator+=** ([decimal64](#) __rhs)
- **decimal64** & **operator+=** (int __rhs)
- **decimal64** & **operator+=** (long __rhs)
- **decimal64** & **operator+=** (long long __rhs)
- **decimal64** & **operator+=** (unsigned int __rhs)
- **decimal64** & **operator+=** (unsigned long __rhs)
- **decimal64** & **operator+=** (unsigned long long __rhs)
- **decimal64** & **operator--** ()
- **decimal64** **operator--** (int)
- **decimal64** & **operator-=** ([decimal128](#) __rhs)
- **decimal64** & **operator-=** ([decimal32](#) __rhs)
- **decimal64** & **operator-=** ([decimal64](#) __rhs)
- **decimal64** & **operator-=** (int __rhs)
- **decimal64** & **operator-=** (long __rhs)
- **decimal64** & **operator-=** (long long __rhs)
- **decimal64** & **operator-=** (unsigned int __rhs)
- **decimal64** & **operator-=** (unsigned long __rhs)
- **decimal64** & **operator-=** (unsigned long long __rhs)
- **decimal64** & **operator/=** ([decimal128](#) __rhs)
- **decimal64** & **operator/=** ([decimal32](#) __rhs)
- **decimal64** & **operator/=** ([decimal64](#) __rhs)
- **decimal64** & **operator/=** (int __rhs)
- **decimal64** & **operator/=** (long __rhs)
- **decimal64** & **operator/=** (long long __rhs)
- **decimal64** & **operator/=** (unsigned int __rhs)
- **decimal64** & **operator/=** (unsigned long __rhs)
- **decimal64** & **operator/=** (unsigned long long __rhs)

6.390.1 Detailed Description

3.2.3 Class decimal64.

6.390.2 Constructor & Destructor Documentation

decimal64()

```
std::decimal::decimal64::decimal64 (
    __decfloat64 __z ) [inline]
```

Conforming extension: Conversion from scalar decimal type.

The documentation for this class was generated from the following file:

- [decimal](#)

6.391 simd_abi::deduce<_Tp, _Np,...> Struct Template Reference

```
#include <simd.h>
```

Inherits `__deduce_impl<_Tp, _Np, typename>`.

6.391.1 Detailed Description

```
template<typename _Tp, size_t _Np, typename...>
```

```
struct simd_abi::deduce<_Tp, _Np,...>
```

Template Parameters

<code>_Tp</code>	The requested <code>value_type</code> for the elements.
<code>_Np</code>	The requested number of elements.
<code>_Abis</code>	This parameter is ignored, since this implementation cannot make any use of it. Either <code>__a</code> good native ABI is matched and used as <code>type</code> alias, or the <code>fixed_size<_Np></code> ABI is used, which internally is built from the best matching native ABIs.

The documentation for this struct was generated from the following file:

- [simd.h](#)

6.392 __gnu_pbds::detail::default_comb_hash_fn Struct Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef [direct_mask_range_hashing](#) `type`

6.392.1 Detailed Description

Primary template, `default_comb_hash_fn`.

6.392.2 Member Typedef Documentation

type

```
typedef direct\_mask\_range\_hashing __gnu_pbds::detail::default_comb_hash_fn::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.393 `std::default_delete<_Tp>` Struct Template Reference

```
#include <unique_ptr.h>
```

Public Member Functions

- constexpr `default_delete` () noexcept=default
- template<typename `_Up`, typename = `_Require<is_convertible<_Up*, _Tp*>>>`
`_GLIBCXX23_CONSTEXPR default_delete` (const `default_delete<_Up>` &) noexcept
- `_GLIBCXX23_CONSTEXPR void operator()` (`_Tp *__ptr`) const

6.393.1 Detailed Description

```
template<typename _Tp>
struct std::default_delete<_Tp>
```

Primary template of `default_delete`, used by `unique_ptr` for single objects.

Since

C++11

6.393.2 Constructor & Destructor Documentation

`default_delete()` [1/2]

```
template<typename _Tp>
constexpr std::default_delete<_Tp>::default_delete ( ) [constexpr], [default], [noexcept]
```

Default constructor.

`default_delete()` [2/2]

```
template<typename _Tp>
template<typename _Up, typename = _Require<is_convertible<_Up*, _Tp*>>>
_GLIBCXX23_CONSTEXPR std::default_delete<_Tp>::default_delete (
    const default_delete<_Up> & ) [inline], [noexcept]
```

Converting constructor.

Allows conversion from a deleter for objects of another type, `_Up`, only if `_Up*` is convertible to `_Tp*`.

6.393.3 Member Function Documentation

`operator>()`

```
template<typename _Tp>
_GLIBCXX23_CONSTEXPR void std::default_delete<_Tp>::operator() (
    _Tp * __ptr ) const [inline]
```

Calls `delete __ptr`

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

6.394 `std::default_delete<_Tp[]>` Struct Template Reference

```
#include <unique_ptr.h>
```


Public Member Functions

- constexpr [default_delete](#) () noexcept=default
- template<typename _Up, typename = _Require<is_convertible<_Up(*)[], _Tp(*)[]>>>
_GLIBCXX23_CONSTEXPR [default_delete](#) (const [default_delete](#)<_Up[]> &) noexcept
- template<typename _Up >
_GLIBCXX23_CONSTEXPR [enable_if](#)<[is_convertible](#)<_Up(*)[], _Tp(*)[]>::value >::type [operator\(\)](#) (_Up *↵
__ptr) const

6.394.1 Detailed Description

template<typename _Tp>
struct std::default_delete<_Tp[]>

Specialization of [default_delete](#) for arrays, used by [unique_ptr](#)<T[]>

6.394.2 Constructor & Destructor Documentation

default_delete() [1/2]

```
template<typename _Tp >
constexpr std::default\_delete<_Tp[]>::default_delete ( ) [constexpr], [default], [noexcept]
```

Default constructor.

default_delete() [2/2]

```
template<typename _Tp >
template<typename _Up, typename = _Require<is_convertible<_Up(*)[], _Tp(*)[]>>>
_GLIBCXX23_CONSTEXPR std::default\_delete<_Tp[]>::default_delete (
    const default\_delete<_Up[]> & ) [inline], [noexcept]
```

Converting constructor.

Allows conversion from a deleter for arrays of another type, such as a const-qualified version of [_Tp](#).

Conversions from types derived from [_Tp](#) are not allowed because it is undefined to [delete\[\]](#) an array of derived types through a pointer to the base type.

6.394.3 Member Function Documentation

operator>()()

```
template<typename _Tp >
template<typename _Up >
_GLIBCXX23_CONSTEXPR enable\_if<is\_convertible<_Up(*)[], _Tp(*)[]>::value >::type std::default\_delete<
_Tp[]>::operator() (
    _Up * __ptr ) const [inline]
```

Calls [delete\[\]](#) [__ptr](#)

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

6.395 __gnu_pbds::detail::default_eq_fn<Key> Struct Template Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef [std::equal_to](#)<Key> [type](#)

6.395.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_eq_fn< Key >
```

Primary template, default_eq_fn.

6.395.2 Member Typedef Documentation

type

```
template<typename Key >
typedef std::equal_to<Key> __gnu_pbds::detail::default_eq_fn< Key >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.396 `__gnu_pbds::detail::default_hash_fn< Key >` Struct Template Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef std::tr1::hash< Key > [type](#)

6.396.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_hash_fn< Key >
```

Primary template, default_hash_fn.

6.396.2 Member Typedef Documentation

type

```
template<typename Key >
typedef std::tr1::hash<Key> __gnu_pbds::detail::default_hash_fn< Key >::type
```

Dispatched type.

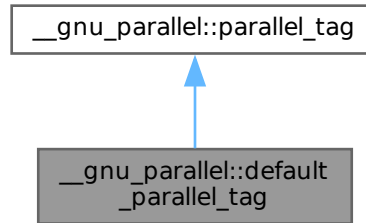
The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.397 `__gnu_parallel::default_parallel_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::default_parallel_tag`:



Public Member Functions

- `default_parallel_tag` ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) `__get_num_threads` ()
- void `set_num_threads` ([_ThreadIndex](#) __num_threads)

6.397.1 Detailed Description

Recommends parallel execution using the default parallel algorithm.

6.397.2 Member Function Documentation

`__get_num_threads()`

[_ThreadIndex](#) `__gnu_parallel::parallel_tag::__get_num_threads` () [inline], [inherited]

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

void `__gnu_parallel::parallel_tag::set_num_threads` ([_ThreadIndex](#) __num_threads) [inline], [inherited]

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.398 `__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >` Struct Template Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef `cond_type::__type` [type](#)

6.398.1 Detailed Description

```
template<typename Comb_Probe_Fn>
struct __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >
```

Primary template, `default_probe_fn`.

6.398.2 Member Typedef Documentation

type

```
template<typename Comb_Probe_Fn >
typedef cond_type::__type __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.399 `__gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >` Struct Template Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef [hash_standard_resize_policy](#)< `size_policy_type`, [trigger](#), `false`, `size_type` > [type](#)

6.399.1 Detailed Description

```
template<typename Comb_Hash_Fn>
struct __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >
```

Primary template, `default_resize_policy`.

6.399.2 Member Typedef Documentation

type

```
template<typename Comb_Hash_Fn >
typedef hash\_standard\_resize\_policy<size_policy_type, trigger, false, size_type> __gnu_pbds::detail::default_res...
```

`Comb_Hash_Fn >::type`

Dispatched type.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.400 `std::default_sentinel_t` Struct Reference

```
#include <iterator_concepts.h>
```

6.400.1 Detailed Description

A sentinel type that can be used to check for the end of a range.

For some iterator types the past-the-end sentinel value is independent of the underlying sequence, and a default sentinel can be used with them. For example, a `std::counted_iterator` keeps a count of how many elements remain, and so checking for the past-the-end value only requires checking if that count has reached zero. A past-the-end `std::istream_iterator` is equal to the default-constructed value, which can be easily checked.

Comparing iterators of these types to `std::default_sentinel` is a convenient way to check if the end has been reached.

Since

C++20

The documentation for this struct was generated from the following file:

- [iterator_concepts.h](#)

6.401 `__gnu_pbds::detail::default_trie_access_traits< Key >` Struct Template Reference

6.401.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_trie_access_traits< Key >
```

Primary template, `default_trie_access_traits`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.402 `__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >` Struct Template Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef [trie_string_access_traits< string_type > type](#)

6.402.1 Detailed Description

```
template<typename Char, typename Char_Traits>
struct __gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator<
char > > >
```

Partial specialization, `default_trie_access_traits`.

6.402.2 Member Typedef Documentation

type

```
template<typename Char , typename Char_Traits >
typedef trie\_string\_access\_traits<string\_type> __gnu_pbds::detail::default_trie_access_traits<
std::basic_string< Char, Char_Traits, std::allocator< char > > >::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

6.403 `__gnu_pbds::detail::default_update_policy` Struct Reference

```
#include <standard_policies.hpp>
```

Public Types

- typedef `lu_move_to_front_policy` type

6.403.1 Detailed Description

Default update policy.

6.403.2 Member Typedef Documentation

type

```
typedef lu_move_to_front_policy __gnu_pbds::detail::default_update_policy::type
```

Dispatched type.

The documentation for this struct was generated from the following file:

- `standard_policies.hpp`

6.404 `std::defer_lock_t` Struct Reference

```
#include <std_mutex.h>
```

6.404.1 Detailed Description

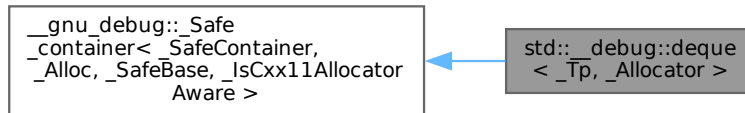
Do not acquire ownership of the mutex.

The documentation for this struct was generated from the following file:

- `std_mutex.h`

6.405 `std::__debug::deque<_Tp, _Allocator>` Class Template Reference

Inheritance diagram for `std::__debug::deque<_Tp, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_const_iterator, deque>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator<const_iterator>` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**

- typedef `__gnu_debug::_Safe_iterator`< `_Base_iterator`, `deque` > `iterator`
- typedef `_Base::pointer` `pointer`
- typedef `_Base::reference` `reference`
- typedef `std::reverse_iterator`< `iterator` > `reverse_iterator`
- typedef `_Base::size_type` `size_type`
- typedef `_Tp` `value_type`

Public Member Functions

- `deque` (`_Base_ref` `__x`)
- template<class `_InputIterator` , typename = `std::_RequireInputIter`<`_InputIterator`>>
`deque` (`_InputIterator` `__first`, `_InputIterator` `__last`, const `_Allocator` & `__a`=`_Allocator`())
- `deque` (const `_Allocator` & `__a`)
- `deque` (const `deque` &)=default
- `deque` (const `deque` & `__d`, const `__type_identity_t`< `_Allocator` > & `__a`)
- `deque` (`deque` &&)=default
- `deque` (`deque` && `__d`, const `__type_identity_t`< `_Allocator` > & `__a`)
- `deque` (`initializer_list`< `value_type` > `__l`, const `allocator_type` & `__a`=`allocator_type`())
- `deque` (`size_type` `__n`, const `__type_identity_t`< `_Tp` > & `__value`, const `_Allocator` & `__a`=`_Allocator`())
- `deque` (`size_type` `__n`, const `_Allocator` & `__a`=`_Allocator`())
- const `_Base` & `_M_base` () const noexcept
- `_Base` & `_M_base` () noexcept
- template<class `_InputIterator` , typename = `std::_RequireInputIter`<`_InputIterator`>>
void `assign` (`_InputIterator` `__first`, `_InputIterator` `__last`)
- void `assign` (`initializer_list`< `value_type` > `__l`)
- void `assign` (`size_type` `__n`, const `_Tp` & `__t`)
- const `reference` `back` () const noexcept
- `reference` `back` () noexcept
- const `iterator` `begin` () const noexcept
- `iterator` `begin` () noexcept
- const `iterator` `cbegin` () const noexcept
- const `iterator` `cend` () const noexcept
- void `clear` () noexcept
- const `reverse_iterator` `crbegin` () const noexcept
- const `reverse_iterator` `crend` () const noexcept
- template<typename... `_Args`>
`iterator` `emplace` (const `iterator` `__position`, `_Args` &&... `__args`)
- template<typename... `_Args`>
`reference` `emplace_back` (`_Args` &&... `__args`)
- template<typename... `_Args`>
`reference` `emplace_front` (`_Args` &&... `__args`)
- const `iterator` `end` () const noexcept
- `iterator` `end` () noexcept
- `iterator` `erase` (const `iterator` `__first`, const `iterator` `__last`)
- `iterator` `erase` (const `iterator` `__position`)
- const `reference` `front` () const noexcept
- `reference` `front` () noexcept
- template<class `_InputIterator` , typename = `std::_RequireInputIter`<`_InputIterator`>>
`iterator` `insert` (const `iterator` `__position`, `_InputIterator` `__first`, `_InputIterator` `__last`)
- `iterator` `insert` (const `iterator` `__position`, `_Tp` && `__x`)
- `iterator` `insert` (const `iterator` `__position`, const `_Tp` & `__x`)

- [iterator insert](#) ([const_iterator](#) __position, [initializer_list](#)< value_type > __l)
- [iterator insert](#) ([const_iterator](#) __position, size_type __n, const _Tp &__x)
- [deque & operator=](#) (const [deque](#) &)=default
- [deque & operator=](#) ([deque](#) &&)=default
- [deque & operator=](#) ([initializer_list](#)< value_type > __l)
- const_reference [operator\[\]](#) (size_type __n) const noexcept
- reference [operator\[\]](#) (size_type __n) noexcept
- void [pop_back](#) () noexcept
- void [pop_front](#) () noexcept
- void [push_back](#) (_Tp &&__x)
- void [push_back](#) (const _Tp &__x)
- void [push_front](#) (_Tp &&__x)
- void [push_front](#) (const _Tp &__x)
- [const_reverse_iterator rbegin](#) () const noexcept
- [reverse_iterator rbegin](#) () noexcept
- [const_reverse_iterator rend](#) () const noexcept
- [reverse_iterator rend](#) () noexcept
- void [resize](#) (size_type __sz)
- void [resize](#) (size_type __sz, const _Tp &__c)
- void [shrink_to_fit](#) () noexcept
- void [swap](#) ([deque](#) &__x) noexcept(*/*conditional */*)

Protected Member Functions

- void [_M_swap](#) (_Safe_container &__x) noexcept

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >
class ::__gnu_debug::_Safe_iterator`

6.405.1 Detailed Description

`template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__debug::deque<_Tp, _Allocator >`

Class std::deque with safety/checking/debug instrumentation.

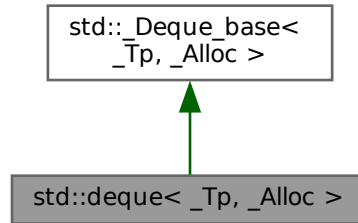
The documentation for this class was generated from the following file:

- [debug/deque](#)

6.406 std::deque<_Tp, _Alloc > Class Template Reference

```
#include <stl_deque.h>
```


Inheritance diagram for `std::deque<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Base::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `_Base::iterator` **iterator**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `deque()`=default
- template<typename `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>>`
`deque` (`_InputIterator` `__first`, `_InputIterator` `__last`, `const allocator_type &__a=allocator_type()`)
- `deque` (`const allocator_type &__a`)
- `deque` (`const deque &__x`)
- `deque` (`const deque &__x`, `const __type_identity_t< allocator_type > &__a`)
- `deque` (`deque &&__x`)=default
- `deque` (`deque &&__x`, `const __type_identity_t< allocator_type > &__a`)
- `deque` (`initializer_list< value_type > __l`, `const allocator_type &__a=allocator_type()`)
- `deque` (`size_type __n`, `const allocator_type &__a=allocator_type()`)
- `deque` (`size_type __n`, `const value_type &__value`, `const allocator_type &__a=allocator_type()`)
- `~deque()`
- template<typename `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>>`
void `assign` (`_InputIterator` `__first`, `_InputIterator` `__last`)
- void `assign` (`initializer_list< value_type > __l`)
- void `assign` (`size_type __n`, `const value_type &__val`)
- reference `at` (`size_type __n`)
- const_reference `at` (`size_type __n`) const

- const_reference [back](#) () const noexcept
- reference [back](#) () noexcept
- const_iterator [begin](#) () const noexcept
- iterator [begin](#) () noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) () noexcept
- const_reverse_iterator [crbegin](#) () const noexcept
- const_reverse_iterator [crend](#) () const noexcept
- template<typename... _Args>
iterator [emplace](#) (const_iterator __position, _Args &&... __args)
- template<typename... _Args>
reference [emplace_back](#) (_Args &&... __args)
- template<typename... _Args>
reference [emplace_front](#) (_Args &&... __args)
- bool [empty](#) () const noexcept
- const_iterator [end](#) () const noexcept
- iterator [end](#) () noexcept
- iterator [erase](#) (const_iterator __first, const_iterator __last)
- iterator [erase](#) (const_iterator __position)
- const_reference [front](#) () const noexcept
- reference [front](#) () noexcept
- allocator_type [get_allocator](#) () const noexcept
- iterator [insert](#) (const_iterator __p, initializer_list< value_type > __l)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
iterator [insert](#) (const_iterator __position, _InputIterator __first, _InputIterator __last)
- iterator [insert](#) (const_iterator __position, const value_type &__x)
- iterator [insert](#) (const_iterator __position, size_type __n, const value_type &__x)
- iterator [insert](#) (const_iterator __position, value_type &&__x)
- size_type [max_size](#) () const noexcept
- deque & [operator=](#) (const deque &__x)
- deque & [operator=](#) (deque &&__x) noexcept(_Alloc_traits::_S_always_equal())
- deque & [operator=](#) (initializer_list< value_type > __l)
- const_reference [operator\[\]](#) (size_type __n) const noexcept
- reference [operator\[\]](#) (size_type __n) noexcept
- void [pop_back](#) () noexcept
- void [pop_front](#) () noexcept
- void [push_back](#) (const value_type &__x)
- void [push_back](#) (value_type &&__x)
- void [push_front](#) (const value_type &__x)
- void [push_front](#) (value_type &&__x)
- const_reverse_iterator [rbegin](#) () const noexcept
- reverse_iterator [rbegin](#) () noexcept
- const_reverse_iterator [rend](#) () const noexcept
- reverse_iterator [rend](#) () noexcept
- void [resize](#) (size_type __new_size)
- void [resize](#) (size_type __new_size, const value_type &__x)
- void [shrink_to_fit](#) () noexcept
- size_type [size](#) () const noexcept
- void [swap](#) (deque &__x) noexcept

Protected Types

- enum { **_S_initial_map_size** }
- typedef [__gnu_cxx::__alloc_traits](#)<_Map_alloc_type > **_Map_alloc_traits**
- typedef _Alloc_traits::template rebind<_Ptr >::other **_Map_alloc_type**
- typedef _Alloc_traits::pointer **_Ptr**
- typedef _Alloc_traits::const_pointer **_Ptr_const**

Protected Member Functions

- template<typename _ForwardIterator >
void **_M_assign_aux** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _InputIterator >
void **_M_assign_aux** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- void **_M_deallocate_map** (_Map_pointer __p, size_t __n) noexcept
- void **_M_default_append** (size_type __n)
- void **_M_default_initialize** ()
- template<typename _Alloc1 >
void **_M_destroy_data** (iterator __first, iterator __last, const _Alloc1 &)
- void **_M_destroy_data** (iterator __first, iterator __last, const [std::allocator](#)<_Tp > &)
- void **_M_destroy_data_aux** (iterator __first, iterator __last)
- [iterator](#) **_M_erase** (iterator __first, iterator __last)
- [iterator](#) **_M_erase** (iterator __pos)
- void **_M_erase_at_begin** (iterator __pos)
- void **_M_erase_at_end** (iterator __pos)
- void **_M_fill_assign** (size_type __n, const value_type &__val)
- void **_M_fill_initialize** (const value_type &__value)
- void **_M_fill_insert** (iterator __pos, size_type __n, const value_type &__x)
- _Map_alloc_type **_M_get_map_allocator** () const noexcept
- template<typename... _Args>
[iterator](#) **_M_insert_aux** (iterator __pos, _Args &&... __args)
- template<typename _ForwardIterator >
void **_M_insert_aux** (iterator __pos, _ForwardIterator __first, _ForwardIterator __last, size_type __n)
- void **_M_insert_aux** (iterator __pos, size_type __n, const value_type &__x)
- void **_M_move_assign1** (deque &&__x, [false_type](#))
- void **_M_move_assign1** (deque &&__x, [true_type](#)) noexcept
- void **_M_move_assign2** (deque &&__x, [false_type](#))
- void **_M_move_assign2** (deque &&__x, [true_type](#))
- void **_M_range_check** (size_type __n) const
- template<typename _ForwardIterator >
void **_M_range_insert_aux** (iterator __pos, _ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _InputIterator >
void **_M_range_insert_aux** (iterator __pos, _InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename... _Args>
void **_M_replace_map** (_Args &&... __args)
- bool **_M_shrink_to_fit** ()
- template<typename _InputIterator >
void **_M_range_initialize** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
void **_M_range_initialize** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))

- `template<typename... _Args>`
`void _M_push_back_aux (_Args &&... __args)`
- `template<typename... _Args>`
`void _M_push_front_aux (_Args &&... __args)`
- `void _M_pop_back_aux ()`
- `void _M_pop_front_aux ()`
- `iterator _M_reserve_elements_at_front (size_type __n)`
- `iterator _M_reserve_elements_at_back (size_type __n)`
- `void _M_new_elements_at_front (size_type __new_elements)`
- `void _M_new_elements_at_back (size_type __new_elements)`
- `void _M_reserve_map_at_back (size_type __nodes_to_add=1)`
- `void _M_reserve_map_at_front (size_type __nodes_to_add=1)`
- `void _M_reallocate_map (size_type __nodes_to_add, bool __add_at_front)`

Static Protected Member Functions

- `static size_t _S_check_init_len (size_t __n, const allocator_type &__a)`
- `static size_type _S_max_size (const _Tp_alloc_type &__a) noexcept`

6.406.1 Detailed Description

`template<typename _Tp, typename _Alloc = std::allocator<_Tp>>`
`class std::deque<_Tp, _Alloc >`

A standard container using fixed-size memory allocation and constant-time manipulation of elements at either end.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#).

In previous HP/SGI versions of deque, there was an extra template parameter so users could control the node size. This extension turned out to violate the C++ standard (it can be detected using template template parameters), and it was removed.

Here's how a `deque<Tp>` manages memory. Each deque has 4 members:

- `Tp** _M_map`
- `size_t _M_map_size`
- `iterator _M_start, _M_finish`

`map_size` is at least 8. `map` is an array of `map_size` pointers-to-*nodes*. (The name `map` has nothing to do with the `std::map` class, and **nodes** should not be confused with `std::list`'s usage of *node*.)

A *node* has no specific type name as such, but it is referred to as *node* in this file. It is a simple array-of-`Tp`. If `Tp` is very large, there will be one `Tp` element per node (i.e., an *array* of one). For non-huge `Tp`'s, node size is inversely related to

Tp size: the larger the Tp, the fewer Tp's will fit in a node. The goal here is to keep the total size of a node relatively small and constant over different Tp's, to improve allocator efficiency.

Not every pointer in the map array will point to a node. If the initial number of elements in the deque is small, the /middle/ map pointers will be valid, and the ones at the edges will be unused. This same situation will arise as the map grows: available map pointers, if any, will be on the ends. As new nodes are created, only a subset of the map's pointers need to be copied *outward*.

Class invariants:

- For any nonsingular iterator i:
 - i.node points to a member of the map array. (Yes, you read that correctly: i.node does not actually point to a node.) The member of the map array is what actually points to the node.
 - i.first == *(i.node) (This points to the node (first Tp element).)
 - i.last == i.first + node_size
 - i.cur is a pointer in the range [i.first, i.last). NOTE: the implication of this is that i.cur is always a dereferenceable pointer, even if i is a past-the-end iterator.
- Start and Finish are always nonsingular iterators. NOTE: this means that an empty deque must have one node, a deque with <N elements (where N is the node buffer size) must have one node, a deque with N through (2N-1) elements must have two nodes, etc.
- For every node other than start.node and finish.node, every element in the node is an initialized object. If start.↔ node == finish.node, then [start.cur, finish.cur) are initialized objects, and the elements outside that range are uninitialized storage. Otherwise, [start.cur, start.last) and [finish.first, finish.cur) are initialized objects, and [start.↔ first, start.cur) and [finish.cur, finish.last) are uninitialized storage.
- [map, map + map_size) is a valid, non-empty range.
- [start.node, finish.node] is a valid range contained within [map, map + map_size).
- A pointer in the range [map, map + map_size) points to an allocated node if and only if the pointer is in the range [start.node, finish.node].

Here's the magic: nothing in deque is **aware** of the discontinuous storage!

The memory setup and layout occurs in the parent, _Base, and the iterator class is entirely responsible for *leaping* from one node to the next. All the implementation routines for deque itself work only through the start and finish iterators. This keeps the routines simple and sane, and we can use other standard algorithms as well.

6.406.2 Constructor & Destructor Documentation

deque() [1/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque< _Tp, _Alloc >::deque ( ) [default]
```

Creates a deque with no elements.

deque() [2/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque< _Tp, _Alloc >::deque (
    const allocator_type & __a ) [inline], [explicit]
```

Creates a deque with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

deque() [3/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque<_Tp, _Alloc >::deque (
    size_type __n,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Creates a deque with default constructed elements.

Parameters

\leftrightarrow __n	The number of elements to initially create.
\leftrightarrow __a	An allocator.

This constructor fills the deque with *n* default constructed elements.

deque() [4/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque<_Tp, _Alloc >::deque (
    size_type __n,
    const value_type & __value,
    const allocator_type & __a = allocator_type() ) [inline]
```

Creates a deque with copies of an exemplar element.

Parameters

__n	The number of elements to initially create.
__value	An element to copy.
__a	An allocator.

This constructor fills the deque with __n copies of __value.

References [std::deque<_Tp, _Alloc >::M_fill_initialize\(\)](#).

deque() [5/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque<_Tp, _Alloc >::deque (
    const deque<_Tp, _Alloc > & __x ) [inline]
```

Deque copy constructor.

Parameters

\leftrightarrow __x	A deque of identical element and allocator types.
--------------------------	---

The newly-created deque uses a copy of the allocator object used by __x (unless the allocator traits dictate a different object).

References [std::deque<_Tp, _Alloc >::begin\(\)](#), and [std::deque<_Tp, _Alloc >::end\(\)](#).

deque() [6/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
```

```
std::deque< _Tp, _Alloc >::deque (
    deque< _Tp, _Alloc > && ) [default]
```

Deque move constructor.

The newly-created deque contains the exact contents of the moved instance. The contents of the moved instance are a valid, but unspecified deque.

deque() [7/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque< _Tp, _Alloc >::deque (
    const deque< _Tp, _Alloc > & __x,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Copy constructor with alternative allocator.

References [std::deque< _Tp, _Alloc >::begin\(\)](#), and [std::deque< _Tp, _Alloc >::end\(\)](#).

deque() [8/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque< _Tp, _Alloc >::deque (
    deque< _Tp, _Alloc > && __x,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Move constructor with alternative allocator.

deque() [9/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque< _Tp, _Alloc >::deque (
    initializer_list< value_type > __l,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a deque from an initializer list.

Parameters

<code>__l</code>	An initializer_list.
<code>__a</code>	An allocator object.

Create a deque consisting of copies of the elements in the initializer_list [`__l`](#).

This will call the element type's copy constructor N times (where N is [`__l.size\(\)`](#)) and do no memory reallocation.

References [std::deque< _Tp, _Alloc >::_M_range_initialize\(\)](#).

deque() [10/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
std::deque< _Tp, _Alloc >::deque (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a deque from a range.

Parameters

<code>__first</code>	An input iterator.
--------------------------------------	--------------------

Parameters

<code>__last</code>	An input iterator.
<code>__a</code>	An allocator object.

Create a deque consisting of copies of the elements from `[__first, __last)`.

If the iterators are forward, bidirectional, or random-access, then this will call the elements' copy constructor *N* times (where *N* is `distance(__first, __last)`) and do no memory reallocation. But if only input iterators are used, then this will do at most *2N* calls to the copy constructor, and *logN* memory reallocations.

References [std::__iterator_category\(\)](#), and [std::deque<_Tp, _Alloc>::__M_range_initialize\(\)](#).

`~deque()`

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::deque<_Tp, _Alloc>::~~deque ( ) [inline]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::deque<_Tp, _Alloc>::begin\(\)](#), and [std::deque<_Tp, _Alloc>::end\(\)](#).

6.406.3 Member Function Documentation`_M_fill_initialize()`

```
template<typename _Tp , typename _Alloc >
void deque::_M_fill_initialize (
    const value_type & __value ) [protected]
```

Fills the deque with copies of `value`.

Parameters

<code>__value</code>	Initial value.
----------------------	----------------

Returns

Nothing.

Precondition

`_M_start` and `_M_finish` have already been initialized, but none of the deque's elements have yet been constructed.

This function is called only when the user provides an explicit size (with or without an explicit exemplar value).

References [std::_Destroy\(\)](#).

Referenced by [std::deque<_Tp, _Alloc>::deque\(\)](#).

`_M_new_elements_at_back()`

```
template<typename _Tp , typename _Alloc >
void deque::_M_new_elements_at_back (
    size_type __new_elements ) [protected]
```

Memory-handling helpers for the previous internal insert functions.

References [std::size\(\)](#).

Referenced by [std::deque<_Tp, _Alloc>::_M_reserve_elements_at_back\(\)](#).

_M_new_elements_at_front()

```
template<typename _Tp , typename _Alloc >
void deque::_M_new_elements_at_front (
    size_type __new_elements ) [protected]
```

Memory-handling helpers for the previous internal insert functions.

References [std::size\(\)](#).

Referenced by [std::deque< _Tp, _Alloc >::_M_reserve_elements_at_front\(\)](#).

_M_pop_back_aux()

```
template<typename _Tp , typename _Alloc >
void deque::_M_pop_back_aux [protected]
```

Helper functions for `push_*` and `pop_*`.

Referenced by [std::deque< _Tp, _Alloc >::pop_back\(\)](#).

_M_pop_front_aux()

```
template<typename _Tp , typename _Alloc >
void deque::_M_pop_front_aux [protected]
```

Helper functions for `push_*` and `pop_*`.

Referenced by [std::deque< _Tp, _Alloc >::pop_front\(\)](#).

_M_push_back_aux()

```
template<typename _Tp , typename _Alloc >
template<typename... _Args>
void deque::_M_push_back_aux (
    _Args &&... __args ) [protected]
```

Helper functions for `push_*` and `pop_*`.

References [std::size\(\)](#).

Referenced by [std::deque< _Tp, _Alloc >::push_back\(\)](#).

_M_push_front_aux()

```
template<typename _Tp , typename _Alloc >
template<typename... _Args>
void deque::_M_push_front_aux (
    _Args &&... __args ) [protected]
```

Helper functions for `push_*` and `pop_*`.

References [std::size\(\)](#).

Referenced by [std::deque< _Tp, _Alloc >::push_front\(\)](#).

_M_range_check()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::_M_range_check (
    size_type __n ) const [inline], [protected]
```

Safety check used only from `at()`.

References [std::deque< _Tp, _Alloc >::size\(\)](#).

Referenced by [std::deque< _Tp, _Alloc >::at\(\)](#), and [std::deque< _Tp, _Alloc >::at\(\)](#).

_M_range_initialize() [1/2]

```
template<typename _Tp , typename _Alloc >
template<typename _ForwardIterator >
```

```
void deque::_M_range_initialize (
    _ForwardIterator __first,
    _ForwardIterator __last,
    std::forward_iterator_tag ) [protected]
```

Fills the deque with whatever is in [first,last).

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

Nothing.

If the iterators are actually forward iterators (or better), then the memory layout can be done all at once. Else we move forward using `push_back` on each value from the iterator.

References [std::_Destroy\(\)](#), [std::advance\(\)](#), and [std::distance\(\)](#).

`_M_range_initialize()` [2/2]

```
template<typename _Tp , typename _Alloc >
template<typename _InputIterator >
void deque::_M_range_initialize (
    _InputIterator __first,
    _InputIterator __last,
    std::input_iterator_tag ) [protected]
```

Fills the deque with whatever is in [first,last).

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

Nothing.

If the iterators are actually forward iterators (or better), then the memory layout can be done all at once. Else we move forward using `push_back` on each value from the iterator.

Referenced by [std::deque< _Tp, _Alloc >::deque\(\)](#), and [std::deque< _Tp, _Alloc >::deque\(\)](#).

`_M_reallocate_map()`

```
template<typename _Tp , typename _Alloc >
void deque::_M_reallocate_map (
    size_type __nodes_to_add,
    bool __add_at_front ) [protected]
```

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

References [std::max\(\)](#).

Referenced by [std::deque< _Tp, _Alloc >::_M_reserve_map_at_back\(\)](#), and [std::deque< _Tp, _Alloc >::_M_reserve_map_at_front\(\)](#).

`_M_reserve_elements_at_back()`

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque< _Tp, _Alloc >::_M_reserve_elements_at_back (
    size_type __n ) [inline], [protected]
```

Memory-handling helpers for the previous internal insert functions.

References [std::deque< _Tp, _Alloc >::_M_new_elements_at_back\(\)](#).

`_M_reserve_elements_at_front()`

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque< _Tp, _Alloc >::_M_reserve_elements_at_front (
    size_type __n ) [inline], [protected]
```

Memory-handling helpers for the previous internal insert functions.

References [std::deque< _Tp, _Alloc >::_M_new_elements_at_front\(\)](#).

`_M_reserve_map_at_back()`

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::_M_reserve_map_at_back (
    size_type __nodes_to_add = 1 ) [inline], [protected]
```

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

References [std::deque< _Tp, _Alloc >::_M_reallocate_map\(\)](#).

`_M_reserve_map_at_front()`

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::_M_reserve_map_at_front (
    size_type __nodes_to_add = 1 ) [inline], [protected]
```

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

References [std::deque< _Tp, _Alloc >::_M_reallocate_map\(\)](#).

`assign()` [1/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
void std::deque< _Tp, _Alloc >::assign (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Assigns a range to a deque.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a deque with copies of the elements in the range `[__first,__last)`.

Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned.






References [std::__iterator_category\(\)](#).

assign() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque<_Tp, _Alloc >::assign (
    initializer_list< value_type > __l ) [inline]
```

Assigns an initializer list to a deque.

Parameters

	An initializer_list.
	
	
	
	

This function fills a deque with copies of the elements in the initializer_list __l.



Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned.

assign() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque<_Tp, _Alloc >::assign (
    size_type __n,
    const value_type & __val ) [inline]
```

Assigns a given value to a deque.

Parameters

	Number of elements to be assigned.
	Value to be assigned.


This function fills a deque with *n* copies of the given value. Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned.

at() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reference std::deque<_Tp, _Alloc >::at (
    size_type __n ) [inline]
```

Provides access to the data contained in the deque.

Parameters

	The index of the element for which data should be accessed.
---	---

Returns

Read/write reference to data.

Exceptions

<i>std::out_of_range</i>	If __n is an invalid index.
--------------------------	-----------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the deque. The function throws `out_of_range` if the check fails.

References [std::deque<_Tp, _Alloc>::_M_range_check\(\)](#).

at() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
const_reference std::deque<_Tp, _Alloc>::at (
    size_type __n ) const [inline]
```

Provides access to the data contained in the deque.

Parameters

<code>__n</code>	The index of the element for which data should be accessed.
------------------	---

Returns

Read-only (constant) reference to data.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the deque. The function throws `out_of_range` if the check fails.

References [std::deque<_Tp, _Alloc>::_M_range_check\(\)](#).

back() [1/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
const_reference std::deque<_Tp, _Alloc>::back ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the last element of the deque.

References [std::deque<_Tp, _Alloc>::end\(\)](#).

back() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
reference std::deque<_Tp, _Alloc>::back ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the last element of the deque.

References [std::deque<_Tp, _Alloc>::end\(\)](#).

begin() [1/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
const_iterator std::deque<_Tp, _Alloc>::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the deque. Iteration is done in ordinary element order.

begin() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
iterator std::deque<_Tp, _Alloc>::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the deque. Iteration is done in ordinary element order.

Referenced by [std::deque< _Tp, _Alloc >::deque\(\)](#), [std::deque< _Tp, _Alloc >::deque\(\)](#), [std::deque< _Tp, _Alloc >::~~deque\(\)](#), [std::deque< _Tp, _Alloc >::clear\(\)](#), [std::deque< _Tp, _Alloc >::front\(\)](#), [std::deque< _Tp, _Alloc >::front\(\)](#), [std::deque< _Tp, _Alloc >::insert\(\)](#), [std::deque< _Tp, _Alloc >::insert\(\)](#), [std::operator<\(\)](#), [std::deque< _Tp, _Alloc >::operator=\(\)](#), and [std::operator==\(\)](#).

cbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::deque< _Tp, _Alloc >::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the deque. Iteration is done in ordinary element order.

Referenced by [std::deque< _Tp, _Alloc >::insert\(\)](#), [std::deque< _Tp, _Alloc >::insert\(\)](#), and [std::deque< _Tp, _Alloc >::insert\(\)](#).

cend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::deque< _Tp, _Alloc >::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the deque. Iteration is done in ordinary element order.

clear()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::deque< _Tp, _Alloc >::begin\(\)](#).

crbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::deque< _Tp, _Alloc >::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

crend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::deque< _Tp, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

emplace()

```
template<typename _Tp , typename _Alloc >
template<typename... _Args>
deque< _Tp, _Alloc >::iterator deque::emplace (
    const_iterator __position,
    _Args &&... __args )
```

Inserts an object in deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type T constructed with T(std::forward<Args>(args)...) before the specified location. Referenced by `std::deque<_Tp, _Alloc>::insert()`.

empty()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
bool std::deque<_Tp, _Alloc>::empty ( ) const [inline], [noexcept]
Returns true if the deque is empty. (Thus begin() would equal end().)
```

end() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::deque<_Tp, _Alloc>::end ( ) const [inline], [noexcept]
Returns a read-only (constant) iterator that points one past the last element in the deque. Iteration is done in ordinary element order.
```

end() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque<_Tp, _Alloc>::end ( ) [inline], [noexcept]
Returns a read/write iterator that points one past the last element in the deque. Iteration is done in ordinary element order.
Referenced by std::deque<_Tp, _Alloc>::deque(), std::deque<_Tp, _Alloc>::deque(), std::deque<_Tp, _Alloc>::~deque(),
std::deque<_Tp, _Alloc>::back(), std::deque<_Tp, _Alloc>::back(), std::operator<(), std::deque<_Tp, _Alloc>::operator=(),
and std::operator==( ).
```

erase() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque<_Tp, _Alloc>::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
Remove a range of elements.
```

Parameters

<code>__first</code>	Iterator pointing to the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by *last* prior to erasing (or end()).

This function will erase the elements in the range [`__first`,`__last`) and shorten the deque accordingly.

The user is cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque<_Tp, _Alloc>::erase (
    const_iterator __position ) [inline]
```

Remove element at given position.

Parameters

<code>__position</code>	Iterator pointing to element to be erased.
-------------------------	--

Returns

An iterator pointing to the next element (or end()).

This function will erase the element at the given position and thus shorten the deque by one.

The user is cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

front() [1/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
const_reference std::deque< _Tp, _Alloc >::front ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the deque.

References [std::deque< _Tp, _Alloc >::begin\(\)](#).

front() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
reference std::deque< _Tp, _Alloc >::front ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the deque.

References [std::deque< _Tp, _Alloc >::begin\(\)](#).

get_allocator()

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
allocator_type std::deque< _Tp, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

Referenced by [std::deque< _Tp, _Alloc >::operator=\(\)](#).

insert() [1/5]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
iterator std::deque< _Tp, _Alloc >::insert (
    const_iterator __p,
    initializer_list< value_type > __l ) [inline]
```

Inserts an initializer list into the deque.

Parameters

<code>__p</code>	An iterator into the deque.
<code>__l</code>	An <code>initializer_list</code> .

Returns

An iterator that points to the inserted data.

This function will insert copies of the data in the `initializer_list __l` into the deque before the location specified by `__p`. This is known as *list insert*.

References [std::deque< _Tp, _Alloc >::begin\(\)](#), and [std::deque< _Tp, _Alloc >::cbegin\(\)](#).

insert() [2/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
iterator std::deque< _Tp, _Alloc >::insert (
    const_iterator __position,
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Inserts a range into the deque.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator that points to the inserted data.

This function will insert copies of the data in the range [`__first`,`__last`) into the deque before the location specified by `__position`. This is known as *range insert*.

References [std::__iterator_category\(\)](#), [std::deque< _Tp, _Alloc >::begin\(\)](#), and [std::deque< _Tp, _Alloc >::cbegin\(\)](#).

insert() [3/5]

```
template<typename _Tp , typename _Alloc >
deque< _Tp, _Alloc >::iterator deque::insert (
    const_iterator __position,
    const value_type & __x )
```

Inserts given value into deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location.

insert() [4/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque< _Tp, _Alloc >::insert (
    const_iterator __position,
    size_type __n,
    const value_type & __x ) [inline]
```

Inserts a number of copies of given data into the deque.

Parameters

<code>__position</code>	A const_iterator into the deque.
-------------------------	----------------------------------

Parameters

<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a specified number of copies of the given data before the location specified by `__position`.
References `std::deque<_Tp, _Alloc>::begin()`, and `std::deque<_Tp, _Alloc>::cbegin()`.

insert() [5/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::deque<_Tp, _Alloc>::insert (
    const_iterator __position,
    value_type && __x ) [inline]
```

Inserts given rvalue into deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location.
References `std::deque<_Tp, _Alloc>::emplace()`, and `std::move()`.

max_size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
size_type std::deque<_Tp, _Alloc>::max_size ( ) const [inline], [noexcept]
```

Returns the size() of the largest possible deque.

operator=() [1/3]

```
template<typename _Tp , typename _Alloc >
deque<_Tp, _Alloc> & deque::operator= (
    const deque<_Tp, _Alloc> & __x )
```

Deque assignment operator.

Parameters

<code>__x</code>	A deque of identical element and allocator types.
------------------	---

All the elements of x are copied.

The newly-created deque uses a copy of the allocator object used by `__x` (unless the allocator traits dictate a different object).

References [std::__addressof\(\)](#), [std::deque<_Tp, _Alloc >::begin\(\)](#), [std::deque<_Tp, _Alloc >::end\(\)](#), [std::deque<_Tp, _Alloc >::get_allocator\(\)](#), [std::deque<_Tp, _Alloc >::size\(\)](#), and [std::size\(\)](#).

operator=() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
deque & std::deque<_Tp, _Alloc >::operator= (
    deque<_Tp, _Alloc > && __x ) [inline], [noexcept]
```

Deque move assignment operator.

Parameters

↩ _X	A deque of identical element and allocator types.
---	---

The contents of [__x](#) are moved into this deque (without copying, if the allocators permit it). [__x](#) is a valid, but unspecified deque.

References [std::move\(\)](#).

operator=() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
deque & std::deque<_Tp, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Assigns an initializer list to a deque.

Parameters

↩ _↩ ↩ _↩ /	An initializer_list.
---	----------------------

This function fills a deque with copies of the elements in the initializer_list [__l](#).

Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned.

operator[]() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reference std::deque<_Tp, _Alloc >::operator[] (
    size_type __n ) const [inline], [noexcept]
```

Subscript access to the data contained in the deque.

Parameters

↩ _n	The index of the element for which data should be accessed.
---	---

Returns

Read-only (constant) reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

operator[]() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reference std::deque< _Tp, _Alloc >::operator[] (
    size_type __n ) [inline], [noexcept]
```

Subscript access to the data contained in the deque.

Parameters

<code>__n</code>	The index of the element for which data should be accessed.
------------------	---

Returns

Read/write reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

pop_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::pop_back ( ) [inline], [noexcept]
```

Removes last element.

This is a typical stack operation. It shrinks the deque by one.

Note that no data is returned, and if the last element's data is needed, it should be retrieved before `pop_back()` is called.

References `std::deque< _Tp, _Alloc >::_M_pop_back_aux()`.

pop_front()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::pop_front ( ) [inline], [noexcept]
```

Removes first element.

This is a typical stack operation. It shrinks the deque by one.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop_front()` is called.

References `std::deque< _Tp, _Alloc >::_M_pop_front_aux()`.

push_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::push_back (
    const value_type & __x ) [inline]
```

Add data to the end of the deque.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the end of the deque and assigns the given data to it. Due to the nature of a deque this operation can be done in constant time.

References [std::deque< _Tp, _Alloc >::_M_push_back_aux\(\)](#).

push_front()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::push_front (
    const value_type & __x ) [inline]
```

Add data to the front of the deque.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the front of the deque and assigns the given data to it. Due to the nature of a deque this operation can be done in constant time.

References [std::deque< _Tp, _Alloc >::_M_push_front_aux\(\)](#).

rbegin() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::deque< _Tp, _Alloc >::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

rbegin() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reverse_iterator std::deque< _Tp, _Alloc >::rbegin ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

rend() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::deque< _Tp, _Alloc >::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

rend() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reverse_iterator std::deque< _Tp, _Alloc >::rend ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

resize() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque< _Tp, _Alloc >::resize (
    size_type __new_size ) [inline]
```

Resizes the deque to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the deque should contain.
-------------------------	--

This function will resize the deque to the specified number of elements. If the number is smaller than the deque's current size the deque is truncated, otherwise default constructed elements are appended.

References [std::deque<_Tp, _Alloc>::size\(\)](#).

resize() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque<_Tp, _Alloc>::resize (
    size_type __new_size,
    const value_type & __x ) [inline]
```

Resizes the deque to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the deque should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the deque to the specified number of elements. If the number is smaller than the deque's current size the deque is truncated, otherwise the deque is extended and new elements are populated with given data.

References [std::deque<_Tp, _Alloc>::size\(\)](#).

shrink_to_fit()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque<_Tp, _Alloc>::shrink_to_fit ( ) [inline], [noexcept]
```

A non-binding request to reduce memory use.

size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
size_type std::deque<_Tp, _Alloc>::size ( ) const [inline], [noexcept]
```

Returns the number of elements in the deque.

Referenced by [std::deque<_Tp, _Alloc>::_M_range_check\(\)](#), [std::deque<_Tp, _Alloc>::operator=\(\)](#), [std::operator==\(, std::deque<_Tp, _Alloc>::resize\(\)](#), and [std::deque<_Tp, _Alloc>::resize\(\)](#).

swap()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::deque<_Tp, _Alloc>::swap (
    deque<_Tp, _Alloc> & __x ) [inline], [noexcept]
```

Swaps data with another deque.

Parameters

<code>__x</code>	A deque of the same element and allocator types.
------------------	--

This exchanges the elements between two deques in constant time. (Four pointers, so it should be quite fast.) Note that

the global `std::swap()` function is specialized such that `std::swap(d1,d2)` will feed to this function. Whether the allocators are swapped depends on the allocator traits. The documentation for this class was generated from the following files:

- [stl_deque.h](#)
- [deque.tcc](#)

6.407 std::destroying_delete_t Struct Reference

6.407.1 Detailed Description

Tag type used to declare a class-specific operator delete that can invoke the destructor before deallocating the memory. The documentation for this struct was generated from the following file:

- [new](#)

6.408 std::tr2::direct_bases< _Tp > Struct Template Reference

Public Types

- typedef [__reflection_typelist](#)< __direct_bases(_Tp)... > **type**

6.408.1 Detailed Description

```
template<typename _Tp>
struct std::tr2::direct_bases< _Tp >
```

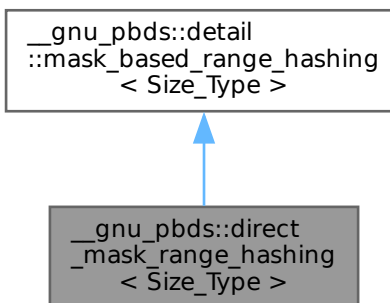
Enumerate all the direct base classes of a class. Form of a typelist. The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

6.409 __gnu_pbds::direct_mask_range_hashing< Size_Type > Class Template Reference

```
#include <hash_policy.hpp>
```

Inheritance diagram for `__gnu_pbds::direct_mask_range_hashing< Size_Type >`:



Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- void **swap** ([direct_mask_range_hashing](#)< Size_Type > &other)

Protected Member Functions

- void **notify_resized** (size_type size)
- size_type [operator\(\)](#) (size_type hash) const
- size_type **range_hash** (size_type hash) const
- void **swap** (mask_based_range_hashing &other)

6.409.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::direct_mask_range_hashing< Size_Type >
```

A mask range-hashing class (uses a bitmask).

6.409.2 Member Function Documentation

operator>()

```
template<typename Size_Type = std::size_t>
size_type __gnu_pbds::direct_mask_range_hashing< Size_Type >::operator() (
    size_type hash ) const [inline], [protected]
```

Transforms the __hash value hash into a ranged-hash value (using a bit-mask).

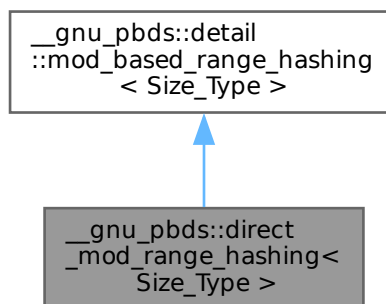
The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.410 __gnu_pbds::direct_mod_range_hashing< Size_Type > Class Template Reference

```
#include <hash_policy.hpp>
```

Inheritance diagram for __gnu_pbds::direct_mod_range_hashing< Size_Type >:



Public Types

- typedef Size_Type **size_type**

Public Member Functions

- void **swap** ([direct_mod_range_hashing](#)< Size_Type > &other)

Protected Member Functions

- void **notify_resized** (size_type size)
- size_type **operator()** (size_type hash) const
- size_type **range_hash** (size_type s) const
- void **swap** (mod_based_range_hashing &other)

6.410.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::direct_mod_range_hashing< Size_Type >
```

A mod range-hashing class (uses the modulo function).

6.410.2 Member Function Documentation**operator>()**

```
template<typename Size_Type = std::size_t>
size_type __gnu_pbds::direct_mod_range_hashing< Size_Type >::operator() (
    size_type hash ) const [inline], [protected]
```

Transforms the __hash value hash into a ranged-hash value (using a modulo operation).

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.411 std::filesystem::directory_entry Class Reference

```
#include <fs_dir.h>
```

Public Member Functions

- **directory_entry** (const [directory_entry](#) &)=default
- **directory_entry** (const [filesystem::path](#) &__p)
- **directory_entry** (const [filesystem::path](#) &__p, [error_code](#) &__ec)
- **directory_entry** ([directory_entry](#) &&) noexcept=default
- void **assign** (const [filesystem::path](#) &__p)
- void **assign** (const [filesystem::path](#) &__p, [error_code](#) &__ec)
- bool **exists** () const
- bool **exists** ([error_code](#) &__ec) const noexcept
- uintmax_t **file_size** () const
- uintmax_t **file_size** ([error_code](#) &__ec) const noexcept
- uintmax_t **hard_link_count** () const
- uintmax_t **hard_link_count** ([error_code](#) &__ec) const noexcept
- bool **is_block_file** () const
- bool **is_block_file** ([error_code](#) &__ec) const noexcept
- bool **is_character_file** () const
- bool **is_character_file** ([error_code](#) &__ec) const noexcept
- bool **is_directory** () const
- bool **is_directory** ([error_code](#) &__ec) const noexcept
- bool **is_fifo** () const

- `bool is_fifo (error_code &__ec) const` noexcept
- `bool is_other () const`
- `bool is_other (error_code &__ec) const` noexcept
- `bool is_regular_file () const`
- `bool is_regular_file (error_code &__ec) const` noexcept
- `bool is_socket () const`
- `bool is_socket (error_code &__ec) const` noexcept
- `bool is_symlink () const`
- `bool is_symlink (error_code &__ec) const` noexcept
- `file_time_type last_write_time () const`
- `file_time_type last_write_time (error_code &__ec) const` noexcept
- `operator const filesystem::path & () const` noexcept
- `bool operator!= (const directory_entry &__rhs) const` noexcept
- `bool operator< (const directory_entry &__rhs) const` noexcept
- `bool operator<= (const directory_entry &__rhs) const` noexcept
- `directory_entry & operator= (const directory_entry &)=default`
- `directory_entry & operator= (directory_entry &&) noexcept=default`
- `bool operator== (const directory_entry &__rhs) const` noexcept
- `bool operator> (const directory_entry &__rhs) const` noexcept
- `bool operator>= (const directory_entry &__rhs) const` noexcept
- `const filesystem::path & path () const` noexcept
- `void refresh ()`
- `void refresh (error_code &__ec) noexcept`
- `void replace_filename (const filesystem::path &__p)`
- `void replace_filename (const filesystem::path &__p, error_code &__ec)`
- `file_status status () const`
- `file_status status (error_code &__ec) const` noexcept
- `file_status symlink_status () const`
- `file_status symlink_status (error_code &__ec) const` noexcept

Friends

- `struct _Dir`
- `class directory_iterator`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const`
`directory_entry &__d)`
- `class recursive_directory_iterator`

6.411.1 Detailed Description

The value type used by directory iterators.

The documentation for this class was generated from the following file:

- [bits/fs_dir.h](#)

6.412 std::filesystem::directory_iterator Class Reference

```
#include <fs_dir.h>
```

Public Types

- typedef ptrdiff_t `difference_type`
- typedef `input_iterator_tag` `iterator_category`
- typedef const `directory_entry` * `pointer`
- typedef const `directory_entry` & `reference`
- typedef `directory_entry` `value_type`

Public Member Functions

- `directory_iterator` (const `directory_iterator` &__rhs)=default
- `directory_iterator` (const `path` &__p)
- `directory_iterator` (const `path` &__p, `directory_options` __options)
- `directory_iterator` (const `path` &__p, `directory_options` __options, `error_code` &__ec)
- `directory_iterator` (const `path` &__p, `error_code` &__ec)
- `directory_iterator` (`directory_iterator` &&__rhs) noexcept=default
- `directory_iterator` & `increment` (`error_code` &__ec)
- const `directory_entry` & `operator*` () const noexcept
- `directory_iterator` & `operator++` ()
- `__directory_iterator_proxy` `operator++` (int)
- const `directory_entry` * `operator->` () const noexcept
- `directory_iterator` & `operator=` (const `directory_iterator` &__rhs)=default
- `directory_iterator` & `operator=` (`directory_iterator` &&__rhs) noexcept=default
- bool `operator==` (`default_sentinel_t`) const noexcept

Friends

- bool `operator==` (const `directory_iterator` &__lhs, const `directory_iterator` &__rhs) noexcept
- class `recursive_directory_iterator`

Related Symbols

(Note that these are not member symbols.)

- `directory_iterator` `begin` (`directory_iterator` __iter) noexcept
- `directory_iterator` `end` (`directory_iterator`) noexcept

6.412.1 Detailed Description

Iterator type for traversing the entries in a single directory.

The documentation for this class was generated from the following file:

- `bits/fs_dir.h`

6.413 `std::discard_block_engine<_RandomNumberEngine, __p, __r>` Class Template Reference

```
#include <random.h>
```

Public Types

- template<typename _Sseq >
using `_If_seed_seq` = typename `enable_if`< __detail::__is_seed_seq< _Sseq, `discard_block_engine`, `result_type` >::value >::type
- typedef `_RandomNumberEngine`::result_type `result_type`

Public Member Functions

- [discard_block_engine](#) ()
- [discard_block_engine](#) (_RandomNumberEngine &&__rng)
- `template<typename _Sseq, typename = _If_seed_seq<_Sseq>>`
[discard_block_engine](#) (_Sseq &__q)
- [discard_block_engine](#) (const _RandomNumberEngine &__rng)
- [discard_block_engine](#) (result_type __s)
- const _RandomNumberEngine & [base](#) () const noexcept
- void [discard](#) (unsigned long long __z)
- [result_type operator\(\)](#) ()
- void [seed](#) ()
- `template<typename _Sseq >`
`_If_seed_seq<_Sseq >` [seed](#) (_Sseq &__q)
- void [seed](#) (result_type __s)

Static Public Member Functions

- static constexpr [result_type max](#) ()
- static constexpr [result_type min](#) ()

Static Public Attributes

- static constexpr size_t [block_size](#)
- static constexpr size_t [used_block](#)

Friends

- `template<typename _RandomNumberEngine1, size_t __p1, size_t __r1, typename _CharT, typename _Traits >`
`std::basic_ostream<_CharT, _Traits > & operator<< (std::basic_ostream<_CharT, _Traits > &__os, const`
`std::discard_block_engine<_RandomNumberEngine1, __p1, __r1 > &__x)`
- `bool operator== (const discard_block_engine &__lhs, const discard_block_engine &__rhs)`
- `template<typename _RandomNumberEngine1, size_t __p1, size_t __r1, typename _CharT, typename _Traits >`
`std::basic_istream<_CharT, _Traits > & operator>> (std::basic_istream<_CharT, _Traits > &__is,`
`std::discard_block_engine<_RandomNumberEngine1, __p1, __r1 > &__x)`

6.413.1 Detailed Description

```
template<typename _RandomNumberEngine, size_t __p, size_t __r>
class std::discard_block_engine<_RandomNumberEngine, __p, __r >
```

Produces random numbers from some base engine by discarding blocks of data.

$0 \leq \text{__r} \leq \text{__p}$

6.413.2 Member Typedef Documentation

result_type

```
template<typename _RandomNumberEngine, size_t __p, size_t __r>
typedef _RandomNumberEngine::result_type std::discard\_block\_engine<_RandomNumberEngine, __p, __r>::result_type
```

The type of the generated random value.

6.413.3 Constructor & Destructor Documentation

discard_block_engine() [1/5]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard_block_engine ( ) [inline]
```

Constructs a default discard_block_engine engine.

The underlying engine is default constructed as well.

discard_block_engine() [2/5]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard_block_engine (
    const _RandomNumberEngine & __rng ) [inline], [explicit]
```

Copy constructs a discard_block_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

discard_block_engine() [3/5]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard_block_engine (
    _RandomNumberEngine && __rng ) [inline], [explicit]
```

Move constructs a discard_block_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

discard_block_engine() [4/5]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard_block_engine (
    result_type __s ) [inline], [explicit]
```

Seed constructs a discard_block_engine engine.

Constructs the underlying generator engine seeded with __s.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

discard_block_engine() [5/5]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard_block_engine (
    _Sseq & __q ) [inline], [explicit]
```

Generator construct a `discard_block_engine` engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

6.413.4 Member Function Documentation

base()

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
const _RandomNumberEngine & std::discard_block_engine< _RandomNumberEngine, __p, __r >::base ( )
const [inline], [noexcept]
```

Gets a const reference to the underlying generator engine object.

discard()

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
void std::discard_block_engine< _RandomNumberEngine, __p, __r >::discard (
    unsigned long long __z ) [inline]
```

Discard a sequence of random numbers.

max()

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
static constexpr result_type std::discard_block_engine< _RandomNumberEngine, __p, __r >::max ( )
[inline], [static], [constexpr]
```

Gets the maximum value in the generated random number range.

min()

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
static constexpr result_type std::discard_block_engine< _RandomNumberEngine, __p, __r >::min ( )
[inline], [static], [constexpr]
```

Gets the minimum value in the generated random number range.

operator()()

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
discard_block_engine< _RandomNumberEngine, __p, __r >::result_type std::discard_block_engine< __q
RandomNumberEngine, __p, __r >::operator()
```

Gets the next value in the generated random number sequence.

seed() [1/3]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
void std::discard_block_engine< _RandomNumberEngine, __p, __r >::seed ( ) [inline]
```

Reseeds the `discard_block_engine` object with the default seed for the underlying base class generator engine.

seed() [2/3]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
template<typename _Sseq >
```

```
_If_seed_seq< _Sseq > std::discard_block_engine< _RandomNumberEngine, __p, __r >::seed (
    _Sseq & __q ) [inline]
```

Reseeds the discard_block_engine object with the given seed sequence.

Parameters

<code>__q</code>	A seed generator function.
------------------	----------------------------

seed() [3/3]

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
void std::discard_block_engine< _RandomNumberEngine, __p, __r >::seed (
    result_type __s ) [inline]
```

Reseeds the discard_block_engine object with the default seed for the underlying base class generator engine.

6.413.5 Friends And Related Symbol Documentation

operator<<

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
template<typename _RandomNumberEngine1 , size_t __p1, size_t __r1, typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::discard_block_engine< _RandomNumberEngine1, __p1, __r1 > & __x ) [friend]
```

Inserts the current state of a discard_block_engine random number generator engine __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A discard_block_engine random number generator engine.

Returns

The output stream with the state of __x inserted or in an error state.

operator==

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
bool operator== (
    const discard_block_engine< _RandomNumberEngine, __p, __r > & __lhs,
    const discard_block_engine< _RandomNumberEngine, __p, __r > & __rhs ) [friend]
```

Compares two discard_block_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A discard_block_engine random number generator object.
<code>__rhs</code>	Another discard_block_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

operator>>

```
template<typename _RandomNumberEngine , size_t __p, size_t __r>
template<typename _RandomNumberEngine1 , size_t __p1, size_t __r1, typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::discard_block_engine< _RandomNumberEngine1, __p1, __r1 > & __x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A discard_block_engine random number generator engine.

Returns

The input stream with the state of __x extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.414 std::discrete_distribution< _IntType > Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _IntType [result_type](#)

Public Member Functions

- template<typename _InputIterator >
discrete_distribution (_InputIterator __wbegin, _InputIterator __wend)
- **discrete_distribution** (const [param_type](#) & __p)
- **discrete_distribution** ([initializer_list](#)< double > __wl)
- template<typename _Func >
discrete_distribution (size_t __nw, double __xmin, double __xmax, _Func __fw)
- template<typename _ForwardIterator , typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator & __urng)

- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `template<typename _UniformRandomNumberGenerator >`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `param_type param () const`
- `void param (const param_type &__param)`
- `std::vector< double > probabilities () const`
- `void reset ()`

Friends

- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::discrete_distribution< _IntType1 > &__x)`
- `bool operator== (const discrete_distribution &__d1, const discrete_distribution &__d2)`
- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::discrete_distribution< _IntType1 > &__x)`

6.414.1 Detailed Description

`template<typename _IntType = int>`
class `std::discrete_distribution< _IntType >`

A `discrete_distribution` random number distribution.
The formula for the discrete probability mass function is

6.414.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::discrete_distribution< _IntType >::result_type
```

The type of the range of the distribution.

6.414.3 Member Function Documentation

max()

```
template<typename _IntType = int>
result_type std::discrete_distribution< _IntType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References `std::vector< _Tp, _Alloc >::empty()`, and `std::vector< _Tp, _Alloc >::size()`.

min()

```
template<typename _IntType = int>
result_type std::discrete_distribution< _IntType >::min ( ) const [inline]
Returns the greatest lower bound value of the distribution.
```

operator()()

```
template<typename _IntType = int>
template<typename _UniformRandomNumberGenerator >
result_type std::discrete_distribution< _IntType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::discrete_distribution< _IntType >::operator\(\)\(\)](#).

Referenced by [std::discrete_distribution< _IntType >::operator\(\)\(\)](#).

param() [1/2]

```
template<typename _IntType = int>
param_type std::discrete_distribution< _IntType >::param ( ) const [inline]
Returns the parameter set of the distribution.
```

param() [2/2]

```
template<typename _IntType = int>
void std::discrete_distribution< _IntType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

probabilities()

```
template<typename _IntType = int>
std::vector< double > std::discrete_distribution< _IntType >::probabilities ( ) const [inline]
Returns the probabilities of the distribution.
References std::vector< \_Tp, \_Alloc >::empty\(\).
```

reset()

```
template<typename _IntType = int>
void std::discrete_distribution< _IntType >::reset ( ) [inline]
Resets the distribution state.
```

6.414.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::discrete_distribution< _IntType1 > & __x ) [friend]
```

Inserts a discrete_distribution random number distribution `___x` into the output stream `___os`.

Parameters

<code>___os</code>	An output stream.
<code>___x</code>	A discrete_distribution random number distribution.

Returns

The output stream with the state of `___x` inserted or in an error state.

operator==

```
template<typename _IntType = int>
bool operator== (
    const discrete_distribution< _IntType > & __d1,
    const discrete_distribution< _IntType > & __d2 ) [friend]
```

Return true if two discrete distributions have the same parameters.

operator>>

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::discrete_distribution< _IntType1 > & __x ) [friend]
```

Extracts a discrete_distribution random number distribution `___x` from the input stream `___is`.

Parameters

<code>___is</code>	An input stream.
<code>___x</code>	A discrete_distribution random number generator engine.

Returns

The input stream with `___x` extracted or in an error state.

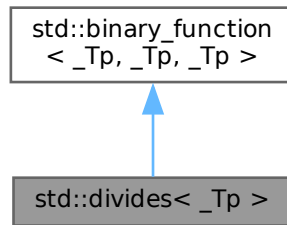
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.415 std::divides< _Tp > Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::divides<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `_Tp` **operator()** (const `_Tp` &`__x`, const `_Tp` &`__y`) const

6.415.1 Detailed Description

template<typename `_Tp`>
struct `std::divides<_Tp>`

One of the [math functors](#).

6.415.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::[first_argument_type](#) [inherited]
[first_argument_type](#) is the type of the first argument

result_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::[result_type](#) [inherited]
[result_type](#) is the return type

second_argument_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::[second_argument_type](#) [inherited]
[second_argument_type](#) is the type of the second argument

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.416 std::divides< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward< _Tp >(__t)/std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t)/std::forward< _Up >(__u))

6.416.1 Detailed Description

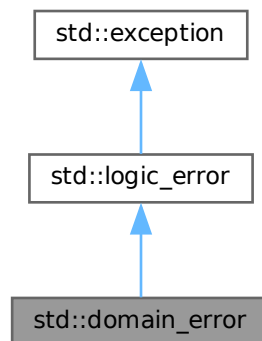
One of the [math functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.417 std::domain_error Class Reference

Inheritance diagram for std::domain_error:



Public Member Functions

- **domain_error** (const char *) _GLIBCXX_TXN_SAFE
- **domain_error** (const [domain_error](#) &)=default
- **domain_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **domain_error** ([domain_error](#) &&)=default
- [domain_error](#) & **operator=** (const [domain_error](#) &)=default
- [domain_error](#) & **operator=** ([domain_error](#) &&)=default
- virtual const char * **what** () const noexcept

6.417.1 Detailed Description

Thrown by the library, or by you, to report domain errors (domain in the mathematical sense).

6.417.2 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

6.418 `__gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >` Struct Template Reference

```
#include <null_node_metadata.hpp>
```

Public Types

- typedef const_iterator **const_reference**
- typedef const_reference **reference**
- typedef const_iterator **value_type**

6.418.1 Detailed Description

```
template<typename Key, typename Data, typename _Alloc>  
struct __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >
```

Constant node iterator.

The documentation for this struct was generated from the following file:

- [null_node_metadata.hpp](#)

6.419 `std::chrono::duration< _Rep, _Period >` Class Template Reference

Public Types

- using **period** = typename _Period::type
- using **rep** = _Rep

Public Member Functions

- template<typename _Rep2 , typename = _Require< is_convertible<const _Rep2&, rep>, __or_<__is_float<rep>, __not_<__is_↵ float<_Rep2>>>>>
constexpr **duration** (const _Rep2 &__rep)
- **duration** (const [duration](#) &)=default
- template<typename _Rep2 , typename _Period2 , typename = _Require< is_convertible<const _Rep2&, rep>, __or_<__is_float<rep>, __and_<__is_harmonic<_Period2>, __not_<__is_float<_Rep2>>>>>>
constexpr **duration** (const [duration](#)< _Rep2, _Period2 > &__d)
- constexpr rep **count** () const

- `template<typename _Rep2 = rep>`
`constexpr enable_if<!treat_as_floating_point<_Rep2>::value, duration & >::type operator%= (const duration &__d)`
- `template<typename _Rep2 = rep>`
`constexpr enable_if<!treat_as_floating_point<_Rep2>::value, duration & >::type operator%= (const rep &__lhs, const duration &__rhs)`
- `constexpr duration & operator*= (const rep &__rhs)`
- `constexpr duration< typename common_type< rep >::type, period > operator+ () const`
- `constexpr duration & operator++ ()`
- `constexpr duration operator++ (int)`
- `constexpr duration & operator+= (const duration &__d)`
- `constexpr duration< typename common_type< rep >::type, period > operator- () const`
- `constexpr duration & operator-- ()`
- `constexpr duration operator-- (int)`
- `constexpr duration & operator-= (const duration &__d)`
- `constexpr duration & operator/= (const rep &__rhs)`
- `duration & operator= (const duration &)=default`

Static Public Member Functions

- static `constexpr duration max () noexcept`
- static `constexpr duration min () noexcept`
- static `constexpr duration zero () noexcept`

Related Symbols

(Note that these are not member symbols.)

- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type operator+ (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, _Rep2 >, _Period > operator* (const duration< _Rep1, _Period > &__lhs, const duration< _Rep2, _Period > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator== (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`

6.419.1 Detailed Description

`template<typename _Rep, typename _Period>`

`class std::chrono::duration<_Rep, _Period>`

`chrono::duration` represents a distance between two points in time

The documentation for this class was generated from the following file:

- [chrono.h](#)

6.420 `std::chrono::duration_values<_Rep>` Struct Template Reference

```
#include <chrono.h>
```


Static Public Member Functions

- static constexpr `_Rep` **max** () noexcept
- static constexpr `_Rep` **min** () noexcept
- static constexpr `_Rep` **zero** () noexcept

6.420.1 Detailed Description

```
template<typename _Rep>
struct std::chrono::duration_values< _Rep >
```

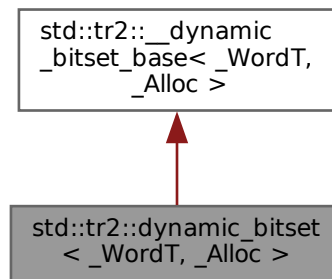
`duration_values`

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.421 `std::tr2::dynamic_bitset< _WordT, _Alloc >` Class Template Reference

Inheritance diagram for `std::tr2::dynamic_bitset< _WordT, _Alloc >`:



Classes

- class [reference](#)

Public Types

- typedef `__dynamic_bitset_base< _WordT, _Alloc >` **_Base**
- typedef `_Alloc` **allocator_type**
- typedef `_WordT` **block_type**
- typedef bool **const_reference**
- typedef `size_t` **size_type**

Public Member Functions

- [dynamic_bitset](#) ()=default
- [dynamic_bitset](#) (const allocator_type &__alloc)
- [dynamic_bitset](#) (const char *__str, const allocator_type &__alloc=allocator_type())
- [dynamic_bitset](#) (const [dynamic_bitset](#) &)=default

- template<typename _CharT, typename _Traits, typename _Alloc1 >
dynamic_bitset (const std::basic_string< _CharT, _Traits, _Alloc1 > &__str, typename basic_string< _CharT, _Traits, _Alloc1 >::size_type __pos=0, typename basic_string< _CharT, _Traits, _Alloc1 >::size_type __n=std::basic_string< _CharT, _Traits, _Alloc1 >::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'), const allocator_type &__alloc=allocator_type())
- dynamic_bitset (dynamic_bitset &&__b) noexcept
- dynamic_bitset (initializer_list< block_type > __il, const allocator_type &__alloc=allocator_type())
- dynamic_bitset (size_type __nbits, unsigned long long __val=0ULL, const allocator_type &__alloc=allocator_type())
- template<typename _Traits = std::char_traits<char>, typename _CharT = typename _Traits::char_type>
void _M_copy_from_ptr (const _CharT *, size_t, size_t, size_t, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
- template<typename _CharT, typename _Traits, typename _Alloc1 >
void _M_copy_from_string (const basic_string< _CharT, _Traits, _Alloc1 > &__str, size_t __pos, size_t __n, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
- template<typename _CharT, typename _Traits, typename _Alloc1 >
void _M_copy_to_string (std::basic_string< _CharT, _Traits, _Alloc1 > &__str, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1')) const
- bool all () const
- bool any () const
- template<typename _BlockInputIterator >
void append (_BlockInputIterator __first, _BlockInputIterator __last)
- void append (block_type __block)
- void append (initializer_list< block_type > __il)
- void clear ()
- size_type count () const noexcept
- bool empty () const noexcept
- size_type find_first () const
- size_type find_next (size_t __prev) const
- dynamic_bitset & flip ()
- dynamic_bitset & flip (size_type __pos)
- allocator_type get_allocator () const noexcept
- bool is_proper_subset_of (const dynamic_bitset &__b) const
- bool is_subset_of (const dynamic_bitset &__b) const
- constexpr size_type max_size () noexcept
- bool none () const
- size_type num_blocks () const noexcept
- dynamic_bitset & operator= (const dynamic_bitset &)=default
- dynamic_bitset & operator= (dynamic_bitset &&__b) noexcept(std::is_nothrow_move_assignable< _Base >::value)
- dynamic_bitset operator~ () const
- void push_back (bool __bit)
- dynamic_bitset & reset ()
- dynamic_bitset & reset (size_type __pos)
- void resize (size_type __nbits, bool __value=false)
- dynamic_bitset & set ()
- dynamic_bitset & set (size_type __pos, bool __val=true)
- size_type size () const noexcept
- void swap (dynamic_bitset &__b) noexcept
- bool test (size_type __pos) const

- `template<typename _CharT = char, typename _Traits = std::char_traits<_CharT>, typename _Alloc1 = std::allocator<_CharT>>>
std::basic_string<_CharT, _Traits, _Alloc1 > to_string (_CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
const`
- `unsigned long long to_ullong () const`
- `unsigned long to_ulong () const`

- `dynamic_bitset & operator&= (const dynamic_bitset &__rhs)`
- `dynamic_bitset & operator&= (dynamic_bitset &&__rhs)`
- `dynamic_bitset & operator|= (const dynamic_bitset &__rhs)`
- `dynamic_bitset & operator^= (const dynamic_bitset &__rhs)`
- `dynamic_bitset & operator-= (const dynamic_bitset &__rhs)`

- `dynamic_bitset & operator<<= (size_type __pos)`
- `dynamic_bitset & operator>>= (size_type __pos)`

- `reference operator[] (size_type __pos)`
- `const_reference operator[] (size_type __pos) const`

- `dynamic_bitset operator<< (size_type __pos) const`
- `dynamic_bitset operator>> (size_type __pos) const`

Static Public Attributes

- static const size_type **bits_per_block**
- static const size_type **npos**

Friends

- bool **operator<** (const dynamic_bitset &__lhs, const dynamic_bitset &__rhs) noexcept
- bool **operator==** (const dynamic_bitset &__lhs, const dynamic_bitset &__rhs) noexcept
- class **reference**

6.421.1 Detailed Description

**template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>>
class std::tr2::dynamic_bitset<_WordT, _Alloc >**

The `dynamic_bitset` class represents a sequence of bits.

See N2050, Proposal to Add a Dynamically Sizeable Bitset to the Standard Library. <http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2006/n2050.pdf>

In the general unoptimized case, storage is allocated in word-sized blocks. Let *B* be the number of bits in a word, then $(Nb+(B-1))/B$ words will be used for storage. *B* - *NbB* bits are unused. (They are the high-order bits in the highest word.) It is a class invariant that those unused bits are always zero.

If you think of `dynamic_bitset` as “a simple array of bits,” be aware that your mental picture is reversed: a `dynamic_bitset` behaves the same way as bits in integers do, with the bit at index 0 in the “least significant / right-hand” position, and the bit at index *Nb*-1 in the “most significant / left-hand” position. Thus, unlike other containers, a `dynamic_bitset`’s index “counts from right to left,” to put it very loosely.

This behavior is preserved when translating to and from strings. For example, the first line of the following program probably prints “b(‘a’) is 0001100001” on a modern ASCII system.

```

#include <dynamic_bitset>
#include <iostream>
#include <sstream>

using namespace std;

int main()
{
    long        a = 'a';
    dynamic_bitset<> b(a);

    cout << "b('a') is " << b << endl;

    ostringstream s;
    s << b;
    string str = s.str();
    cout << "index 3 in the string is " << str[3] << " but\n"
         << "index 3 in the bitset is " << b[3] << endl;
}

```

Most of the actual code isn't contained in `dynamic_bitset<>` itself, but in the base class `__dynamic_bitset_base`. The base class works with whole words, not with individual bits. This allows us to specialize `__dynamic_bitset_base` for the important special case where the `dynamic_bitset` is only a single word.

Extra confusion can result due to the fact that the storage for `__dynamic_bitset_base` is a vector, and is indexed as such. This is carefully encapsulated.

6.421.2 Constructor & Destructor Documentation

`dynamic_bitset()` [1/7]

```

template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset<_WordT, _Alloc >::dynamic_bitset ( ) [default]

```

All bits set to zero.

`dynamic_bitset()` [2/7]

```

template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset<_WordT, _Alloc >::dynamic_bitset (
    const allocator_type & __alloc ) [inline], [explicit]

```

All bits set to zero.

`dynamic_bitset()` [3/7]

```

template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset<_WordT, _Alloc >::dynamic_bitset (
    size_type __nbits,
    unsigned long long __val = 0ULL,
    const allocator_type & __alloc = allocator_type() ) [inline], [explicit]

```

Initial bits bitwise-copied from a single word (others set to zero).

`dynamic_bitset()` [4/7]

```

template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
template<typename _CharT , typename _Traits , typename _Alloc1 >
std::tr2::dynamic_bitset<_WordT, _Alloc >::dynamic_bitset (
    const std::basic_string<_CharT, _Traits, _Alloc1 > & __str,
    typename basic_string<_CharT, _Traits, _Alloc1 >::size_type __pos = 0,
    typename basic_string<_CharT, _Traits, _Alloc1 >::size_type __n = std::basic_string<←
    _CharT, _Traits, _Alloc1>::npos,
    _CharT __zero = _CharT('0'),
    _CharT __one = _CharT('1'),
    const allocator_type & __alloc = allocator_type() ) [inline], [explicit]

```

Use a subset of a string.

Parameters

<code>__str</code>	A string of '0' and '1' characters.
<code>__pos</code>	Index of the first character in <code>__str</code> to use.
<code>__n</code>	The number of characters to copy.
<code>__zero</code>	The character to use for unset bits.
<code>__one</code>	The character to use for set bits.
<code>__alloc</code>	An allocator.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of <code>__str</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither '0' nor '1'.

dynamic_bitset() [5/7]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset (
    const char * __str,
    const allocator_type & __alloc = allocator_type() ) [inline], [explicit]
```

Construct from a string.

Parameters

<code>__str</code>	A string of '0' and '1' characters.
<code>__alloc</code>	An allocator.

Exceptions

<code>std::invalid_argument</code>	If a character appears in the string which is neither '0' nor '1'.
------------------------------------	--

dynamic_bitset() [6/7]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset (
    const dynamic_bitset< _WordT, _Alloc > & ) [default]
```

Copy constructor.

dynamic_bitset() [7/7]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset (
    dynamic_bitset< _WordT, _Alloc > && __b ) [inline], [noexcept]
```

Move constructor.

6.421.3 Member Function Documentation

all()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
bool std::tr2::dynamic_bitset< _WordT, _Alloc >::all ( ) const [inline]
```

Tests whether all the bits are on.

Returns

True if all the bits are set.

any()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
bool std::tr2::dynamic_bitset< _WordT, _Alloc >::any ( ) const [inline]
```

Tests whether any of the bits are on.

Returns

True if at least one bit is set.

append() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
template<typename _BlockInputIterator >
void std::tr2::dynamic_bitset< _WordT, _Alloc >::append (
    _BlockInputIterator __first,
    _BlockInputIterator __last ) [inline]
```

Append an iterator range of blocks.

append() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
void std::tr2::dynamic_bitset< _WordT, _Alloc >::append (
    block_type __block ) [inline]
```

Append a block.

clear()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
void std::tr2::dynamic_bitset< _WordT, _Alloc >::clear ( ) [inline]
```

Clear the bitset.

count()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::count ( ) const [inline], [noexcept]
```

Returns the number of bits which are set.

empty()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
bool std::tr2::dynamic_bitset< _WordT, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the dynamic_bitset is empty.

find_first()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::find_first ( ) const [inline]
```

Finds the index of the first “on” bit.

Returns

The index of the first bit set, or size() if not found.

See also

[find_next](#)

find_next()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::find_next (
    size_t __prev ) const [inline]
```

Finds the index of the next “on” bit after prev.

Returns

The index of the next bit set, or size() if not found.

Parameters

<code>__prev</code>	Where to start searching.
---------------------	---------------------------

See also

[find_first](#)

flip() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::flip ( ) [inline]
```

Toggles every bit to its opposite value.

flip() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::flip (
    size_type __pos ) [inline]
```

Toggles a given bit to its opposite value.

Parameters

<code>__pos</code>	The index of the bit.
--------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

get_allocator()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
allocator_type std::tr2::dynamic_bitset< _WordT, _Alloc >::get_allocator ( ) const [inline],
[noexcept]
```

Return the allocator for the bitset.

max_size()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
constexpr size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::max_size ( ) [inline], [constexpr],
[noexcept]
```

Returns the maximum size of a dynamic_bitset object having the same type as *this. The real answer is max() * bits_per_block but is likely to overflow.

none()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
bool std::tr2::dynamic_bitset< _WordT, _Alloc >::none ( ) const [inline]
```

Tests whether any of the bits are on.

Returns

True if none of the bits are set.

num_blocks()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::num_blocks ( ) const [inline], [noexcept]
```

Returns the total number of blocks.

operator&=() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator&= (
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

operator&=() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator&= (
    dynamic_bitset< _WordT, _Alloc > && __rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

operator-=()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator-= (
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

operator<<()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset std::tr2::dynamic_bitset< _WordT, _Alloc >::operator<< (
    size_type __pos ) const [inline]
```

Self-explanatory.

operator<<=()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator<<= (
    size_type __pos ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__pos</code>	The number of places to shift.
--------------------	--------------------------------

These should be self-explanatory.

operator=() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator= (
    const dynamic_bitset< _WordT, _Alloc > & ) [default]
```

Copy assignment operator.

operator=() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator= (
    dynamic_bitset< _WordT, _Alloc > && __b ) [inline], [noexcept]
```

Move assignment operator.

operator>>()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset std::tr2::dynamic_bitset< _WordT, _Alloc >::operator>> (
    size_type __pos ) const [inline]
```

Self-explanatory.

operator>>=()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator>>= (
    size_type __pos ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__pos</code>	The number of places to shift.
--------------------	--------------------------------

These should be self-explanatory.

operator[]() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
reference std::tr2::dynamic_bitset< _WordT, _Alloc >::operator[] (
    size_type __pos ) [inline]
```

Array-indexing support.

Parameters

<code>__pos</code>	Index into the dynamic_bitset.
--------------------	--------------------------------

Returns

A bool for a 'const dynamic_bitset'. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

operator[]() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
const_reference std::tr2::dynamic_bitset< _WordT, _Alloc >::operator[] (
    size_type __pos ) const [inline]
```

Array-indexing support.

Parameters

<code>__pos</code>	Index into the dynamic_bitset.
--------------------	--------------------------------

Returns

A bool for a 'const dynamic_bitset'. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

operator^=()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator^= (
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

operator" |=()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::operator|= (
    const dynamic_bitset< _WordT, _Alloc > & __rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

operator~()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset std::tr2::dynamic_bitset< _WordT, _Alloc >::operator~ ( ) const [inline]
```

See the no-argument flip().

push_back()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
void std::tr2::dynamic_bitset< _WordT, _Alloc >::push_back (
    bool __bit ) [inline]
```

Push a bit onto the high end of the bitset.

reset() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::reset ( ) [inline]
```

Sets every bit to false.

reset() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::reset (
    size_type __pos ) [inline]
```

Sets a given bit to false.

Parameters

<code>__pos</code>	The index of the bit.
--------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

Same as writing `set (__pos, false)`.

resize()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
void std::tr2::dynamic_bitset< _WordT, _Alloc >::resize (
    size_type __nbits,
    bool __value = false ) [inline]
```

Resize the bitset.

Referenced by [std::tr2::operator>>\(\)](#).

set() [1/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::set ( ) [inline]
```

Sets every bit to true.

set() [2/2]

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset & std::tr2::dynamic_bitset< _WordT, _Alloc >::set (
    size_type __pos,
    bool __val = true ) [inline]
```

Sets a given bit to a particular value.

Parameters

<code>__pos</code>	The index of the bit.
<code>__val</code>	Either true or false, defaults to true.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

size()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
size_type std::tr2::dynamic_bitset< _WordT, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the total number of bits.

Referenced by [std::tr2::operator>>\(\)](#).

swap()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
void std::tr2::dynamic_bitset< _WordT, _Alloc >::swap (
    dynamic_bitset< _WordT, _Alloc > & __b ) [inline], [noexcept]
```

Swap with another bitset.

test()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
bool std::tr2::dynamic_bitset< _WordT, _Alloc >::test (
    size_type __pos ) const [inline]
```

Tests the value of a bit.

Parameters

<code>__pos</code>	The index of a bit.
--------------------	---------------------

Returns

The value at `__pos`.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

to_string()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
template<typename _CharT = char, typename _Traits = std::char_traits<_CharT>, typename _Alloc1 =
std::allocator<_CharT>>
std::basic_string< _CharT, _Traits, _Alloc1 > std::tr2::dynamic_bitset< _WordT, _Alloc >::to_↵
string (
    _CharT __zero = _CharT('0'),
    _CharT __one = _CharT('1') ) const [inline]
```

Returns a character interpretation of the `dynamic_bitset`.

Returns

The string equivalent of the bits.

Note the ordering of the bits: decreasing character positions correspond to increasing bit positions (see the main class notes for an example).

to_ullong()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
unsigned long long std::tr2::dynamic_bitset< _WordT, _Alloc >::to_ullong ( ) const [inline]
```

Returns a numerical interpretation of the `dynamic_bitset`.

Returns

The integral equivalent of the bits.

Exceptions

<code>std::overflow_error</code>	If there are too many bits to be represented in an unsigned long.
----------------------------------	---

to_ulong()

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
unsigned long std::tr2::dynamic_bitset<_WordT, _Alloc>::to_ulong ( ) const [inline]
Returns a numerical interpretation of the dynamic_bitset.
```

Returns

The integral equivalent of the bits.

Exceptions

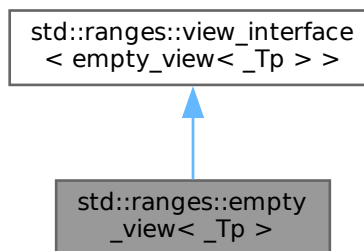
<code>std::overflow_error</code>	If there are too many bits to be represented in an unsigned long.
----------------------------------	---

The documentation for this class was generated from the following files:

- [dynamic_bitset](#)
- [dynamic_bitset.tcc](#)

6.422 std::ranges::empty_view<_Tp> Class Template Reference

Inheritance diagram for `std::ranges::empty_view<_Tp>`:

**Public Member Functions**

- `constexpr decltype(auto) back ()`
- `constexpr decltype(auto) back () const`
- `constexpr auto data () const noexcept(noexcept(ranges::begin(_M_derived())))`
- `constexpr bool empty () const noexcept(noexcept(_S_empty(_M_derived())))`
- `constexpr decltype(auto) front ()`
- `constexpr decltype(auto) front () const`
- `constexpr operator bool () const noexcept(noexcept(ranges::empty(_M_derived())))`

- constexpr **operator bool** () noexcept(noexcept(ranges::empty(_M_derived())))
- constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n)
- constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n) const
- constexpr auto **size** () const noexcept(noexcept(_S_size(_M_derived())))

Static Public Member Functions

- static constexpr _Tp * **begin** () noexcept
- static constexpr _Tp * **data** () noexcept
- static constexpr bool **empty** () noexcept
- static constexpr _Tp * **end** () noexcept
- static constexpr size_t **size** () noexcept

6.422.1 Detailed Description

```
template<typename _Tp>
requires is_object_v<_Tp>
class std::ranges::empty_view<_Tp>
```

A view that contains no elements.

The documentation for this class was generated from the following file:

- [ranges](#)

6.423 `std::enable_if< bool, _Tp >` Struct Template Reference

6.423.1 Detailed Description

```
template<bool, typename _Tp = void>
struct std::enable_if< bool, _Tp >
```

Define a member typedef `type` only if a boolean constant is true.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.424 `std::enable_shared_from_this< _Tp >` Class Template Reference

```
#include <memory>
```

Public Member Functions

- [shared_ptr](#)< _Tp > **shared_from_this** ()
- [shared_ptr](#)< const _Tp > **shared_from_this** () const
- [weak_ptr](#)< _Tp > [weak_from_this](#) () noexcept
- [weak_ptr](#)< const _Tp > [weak_from_this](#) () const noexcept

Protected Member Functions

- **enable_shared_from_this** (const [enable_shared_from_this](#) &) noexcept
- [enable_shared_from_this](#) & **operator=** (const [enable_shared_from_this](#) &) noexcept

Friends

- const [enable_shared_from_this](#) * [__enable_shared_from_this_base](#) (const [__shared_count](#)<> &, const [enable_shared_from_this](#) *__p)
- template<typename , _Lock_policy >
class [__shared_ptr](#)

6.424.1 Detailed Description

```
template<typename _Tp>  
class std::enable_shared_from_this< _Tp >
```

Base class allowing use of the member function `shared_from_this`.

Since

C++11

6.424.2 Member Function Documentation

weak_from_this() [1/2]

```
template<typename _Tp >  
weak\_ptr< const _Tp > std::enable\_shared\_from\_this< _Tp >::weak_from_this ( ) const [inline],  
[noexcept]
```

Get a `weak_ptr` referring to the object that has `*this` as its base.

Since

C++17

weak_from_this() [2/2]

```
template<typename _Tp >  
weak\_ptr< _Tp > std::enable\_shared\_from\_this< _Tp >::weak_from_this ( ) [inline], [noexcept]
```

Get a `weak_ptr` referring to the object that has `*this` as its base.

Since

C++17

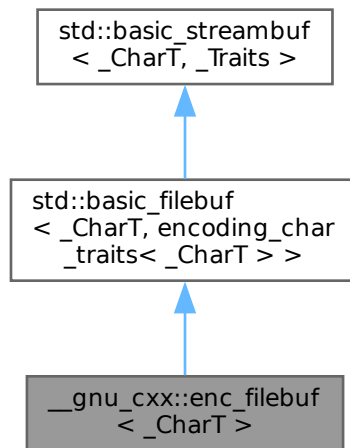
The documentation for this class was generated from the following file:

- [bits/shared_ptr.h](#)

6.425 [__gnu_cxx::enc_filebuf](#)<_CharT > Class Template Reference

```
#include <enc_filebuf.h>
```

Inheritance diagram for `__gnu_cxx::enc_filebuf<_CharT>`:



Public Types

- typedef `codecvt< char_type, char, __state_type >` **__codecvt_type**
- typedef `__basic_file< char >` **__file_type**
- typedef `basic_filebuf< char_type, traits_type >` **filebuf_type**
- typedef `traits_type::state_type` **__state_type**
- typedef `basic_streambuf< char_type, traits_type >` **__streambuf_type**
- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `traits_type::state_type` **state_type**
- typedef `encoding_char_traits< _CharT >` **traits_type**

Public Member Functions

- **enc_filebuf** (`state_type &__state`)
- `__filebuf_type * close` ()
- locale `getloc` () const
- streamsize `in_avail` ()
- bool `is_open` () const throw ()
- `_If_fs_path< _Path, __filebuf_type * >` `open` (const `_Path &__s`, `ios_base::openmode __mode`)
- `__filebuf_type * open` (const `char *__s`, `ios_base::openmode __mode`)
- `__filebuf_type * open` (const `std::string &__s`, `ios_base::openmode __mode`)
- locale `pubimbue` (const locale &`__loc`)
- int_type `sbumpc` ()
- int_type `sgetc` ()
- streamsize `sgetn` (`char_type *__s`, streamsize `__n`)

- int_type [snextc](#) ()
- int_type [sputbackc](#) (char_type __c)
- int_type [sputc](#) (char_type __c)
- streamsize [sputn](#) (const char_type *__s, streamsize __n)
- int_type [sungetc](#) ()
- void [swap](#) ([basic_filebuf](#) &)

- [basic_streambuf](#) * [pubsetbuf](#) (char_type *__s, streamsize __n)
- [pos_type](#) [pubseekoff](#) (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- [pos_type](#) [pubseekpos](#) ([pos_type](#) __sp, ios_base::openmode __mode=ios_base::in|ios_base::out)
- int [pubsync](#) ()

Protected Member Functions

- void [__safe_gbump](#) (streamsize __n)
- void [__safe_pbump](#) (streamsize __n)
- void [_M_allocate_internal_buffer](#) ()
- bool [_M_convert_to_external](#) (char_type *, streamsize)
- void [_M_create_pback](#) ()
- void [_M_destroy_internal_buffer](#) () throw ()
- void [_M_destroy_pback](#) () throw ()
- int [_M_get_ext_pos](#) (__state_type &__state)
- [pos_type](#) [_M_seek](#) (off_type __off, ios_base::seekdir __way, __state_type __state)
- void [_M_set_buffer](#) (streamsize __off)
- bool [_M_terminate_output](#) ()
- void [gbump](#) (int __n)
- virtual void [imbue](#) (const locale &__loc)
- virtual int_type [overflow](#) (int_type __c=_Traits::eof())
- virtual int_type [pbackfail](#) (int_type __c=_Traits::eof())
- void [pbump](#) (int __n)
- virtual [pos_type](#) [seekoff](#) (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual [pos_type](#) [seekpos](#) ([pos_type](#) __pos, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual [__streambuf_type](#) * [setbuf](#) (char_type *__s, streamsize __n)
- void [setg](#) (char_type *__gbeg, char_type *__gnext, char_type *__gend)
- void [setp](#) (char_type *__pbeg, char_type *__pend)
- virtual streamsize [showmanyc](#) ()
- void [swap](#) ([basic_streambuf](#) &__sb)
- virtual int [sync](#) ()
- virtual int_type [uflow](#) ()
- virtual int_type [underflow](#) ()
- virtual streamsize [xsgetn](#) (char_type *__s, streamsize __n)
- virtual streamsize [xspn](#) (const char_type *__s, streamsize __n)

- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const

- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- `char_type * _M_buf`
 - `bool _M_buf_allocated`
 - `locale _M_buf_locale`
 - `size_t _M_buf_size`
 - `const __codecvt_type * _M_codecvt`
 - `char * _M_ext_buf`
 - `streamsize _M_ext_buf_size`
 - `char * _M_ext_end`
 - `const char * _M_ext_next`
 - `__file_type _M_file`
 - `char_type * _M_in_beg`
 - `char_type * _M_in_cur`
 - `char_type * _M_in_end`
 - `__c_lock _M_lock`
 - `ios_base::openmode _M_mode`
 - `char_type * _M_out_beg`
 - `char_type * _M_out_cur`
 - `char_type * _M_out_end`
 - `bool _M_reading`
 - `__state_type _M_state_beg`
 - `__state_type _M_state_cur`
 - `__state_type _M_state_last`
 - `bool _M_writing`
-
- `char_type _M_pback`
 - `char_type * _M_pback_cur_save`
 - `char_type * _M_pback_end_save`
 - `bool _M_pback_init`

6.425.1 Detailed Description

`template<typename _CharT>`
`class __gnu_cxx::enc_filebuf<_CharT>`

class `enc_filebuf`.

6.425.2 Member Function Documentation**`_M_create_pback()`**

`void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_create_pback () [inline], [protected], [inherited]`

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

`_M_destroy_pback()`

`void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_destroy_pback () throw () [inline], [protected], [inherited]`

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

_M_set_buffer()

```
void std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_set_buffer (
    streamsize __off ) [inline], [protected], [inherited]
```

This function sets the pointers of the internal buffer, both get and put areas. Typically:

__off == egptr() - eback() upon underflow/uflow (**read** mode); __off == 0 upon overflow (**write** mode); __off == -1 upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: epptr() - pbase() == _M_buf_size - 1, since _M_buf_size reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

close()

```
basic_filebuf< _CharT, encoding_char_traits< _CharT > >::__filebuf_type * std::basic_filebuf< ↵
    _CharT, encoding_char_traits< _CharT > >::close [inherited]
```

Closes the currently associated file.

Returns

this on success, NULL on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

eback()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::egptr ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow()`.

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic\_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic\_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow\(\)](#).

imbue()

```
void std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::imbue (
    const locale & __loc ) [protected], [virtual], [inherited]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

in_avail()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::in_avail ( ) [inline], [inherited]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

is_open()

```
bool std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::is_open ( ) const throw ( )
[inline], [inherited]
```

Returns true if the external file is open.

open() [1/3]

```
_If_fs_path< _Path, __filebuf_type * > std::basic_filebuf< _CharT, encoding_char_traits< _CharT
> >::open (
    const _Path & __s,
    ios_base::openmode __mode ) [inline], [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a filesystem::path.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

open() [2/3]

```
basic_filebuf<_CharT, encoding_char_traits<_CharT> >::__filebuf_type * std::basic_filebuf<_CharT, encoding_char_traits<_CharT> >::open (
    const char * __s,
    ios_base::openmode __mode ) [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines app, in|out|app, in|app, binary|app, binary|in|out|app, and binary|in|app per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r
	+	+			r+
	+	+	+		w+
	+	+		+	a+
	+			+	a+
+		+			wb
+		+		+	ab
+				+	ab
+		+	+		wb
+	+				rb
+	+	+			r+b
+	+	+	+		w+b
+	+	+		+	a+b
+	+			+	a+b

open() [3/3]

```
__filebuf_type * std::basic_filebuf<_CharT, encoding_char_traits<_CharT> >::open (
    const std::string & __s,
    ios_base::openmode __mode ) [inline], [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

overflow()

```
basic_filebuf< _CharT, encoding_char_traits< _CharT > >::int_type std::basic_filebuf< _CharT,
encoding_char_traits< _CharT > >::overflow (
    int_type __c = _Traits::eof() ) [protected], [virtual], [inherited]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

pbackfail()

```
basic_filebuf< _CharT, encoding_char_traits< _CharT > >::int_type std::basic_filebuf< _CharT,
encoding_char_traits< _CharT > >::pbackfail (
    int_type __c = _Traits::eof() ) [protected], [virtual], [inherited]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (  

    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (  

    const locale & __loc ) [inline], [inherited]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekoff` function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual `sync` function.

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

sbumpc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Referenced by `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<char>::ignore()`, `std::istreambuf_iterator<_CharT, _Traits>::operator++()`, and `std::basic_istream<char>::seekg()`.

seekoff()

```
basic_filebuf<_CharT, encoding_char_traits<_CharT>>::pos_type
std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::seekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

seekpos()

```
basic_filebuf<_CharT, encoding_char_traits<_CharT>>::pos_type
std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::seekpos (
    pos_type __pos,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

setbuf()

```
basic_filebuf<_CharT, encoding_char_traits<_CharT>>::__streambuf_type *
std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::setbuf (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

setg()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert()`.

setp()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setp (
    char_type * __pbeg,
    char_type * __pend) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert()`.

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::istreambuf_iterator<_CharT, _Traits>::operator++()`, and `std::basic_istream<char>::seekg()`.

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for `xsggetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsggetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
streamsize std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::showmanyc [protected],
[virtual], [inherited]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to underflow or uflow] will not return eof() but that they will return immediately.*

The standard adds that *the morphemes of showmanyc are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

snextc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::snextc ( ) [inline], [inherited]
Getting the next character.
```

Returns

The next character, or eof.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::seekg()`, and `std::basic_istream<char>::unget()`.

sputbackc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sputbackc (
    char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(↵__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sputn (
    const char_type * __s,
    streamsize __n) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>↵__s</code>	A buffer read area.
<code>↵__n</code>	A count.

One of two public output functions.

Returns `xsputn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

sungetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sungetc ( ) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

sync()

```
int std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::sync [protected], [virtual],
[inherited]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

uflow()

```
template<typename _CharT, typename _Traits>
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual],
[inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`.

underflow()

```
basic_filebuf< _CharT, encoding_char_traits< _CharT > >::int_type std::basic_filebuf< _CharT,
encoding_char_traits< _CharT > >::underflow [protected], [virtual], [inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

xsgetn()

```
streamsize std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::xsgetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.
Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

`xspn()`

```
streamsize std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::xspn (
    const char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

6.425.3 Member Data Documentation

`_M_buf`

```
char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_buf [protected],
[inherited]
```

Pointer to the beginning of internal buffer.

`_M_buf_locale`

```
template<typename _CharT, typename _Traits>
locale std::basic_streambuf<_CharT, _Traits>::_M_buf_locale [protected], [inherited]
```

Current locale setting.

Referenced by [std::basic_filebuf<_CharT, _Traits>::basic_filebuf\(\)](#).

`_M_buf_size`

```
size_t std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_buf_size [protected],
[inherited]
```

Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.

`_M_ext_buf`

```
char* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_ext_buf [protected],
[inherited]
```

Buffer for external characters. Used for input when `codecvt::always_noconv() == false`. When valid, this corresponds to `eback()`.

_M_ext_buf_size

```
streamsize std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_ext_buf_size [protected],  
[inherited]
```

Size of buffer held by `_M_ext_buf`.

_M_ext_next

```
const char* std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_ext_next [protected],  
[inherited]
```

Pointers into the buffer held by `_M_ext_buf` that delimit a subsequence of bytes that have been read but not yet converted. When valid, `_M_ext_next` corresponds to `egptr()`.

_M_in_beg

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg [protected], [inherited]
```

Start of get area.

_M_in_cur

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur [protected], [inherited]
```

Current read area.

_M_in_end

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end [protected], [inherited]
```

End of get area.

_M_mode

```
ios_base::openmode std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_mode [protected],  
[inherited]
```

Place to stash in || out || in | out settings for current filebuf.

_M_out_beg

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg [protected], [inherited]
```

Start of put area.

_M_out_cur

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur [protected], [inherited]
```

Current put area.

_M_out_end

```
template<typename _CharT , typename _Traits >  
char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end [protected], [inherited]
```

End of put area.

`_M_pback`

```
char_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback [protected],  
[inherited]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

`_M_pback_cur_save`

```
char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_cur_save [protected],  
[inherited]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

`_M_pback_end_save`

```
char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_end_save [protected],  
[inherited]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

`_M_pback_init`

```
bool std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_init [protected],  
[inherited]
```

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

`_M_reading`

```
bool std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_reading [protected],  
[inherited]
```

`_M_reading == false` && `_M_writing == false` for **uncommitted** mode; `_M_reading == true` for **read** mode; `_M_writing == true` for **write** mode;

NB: `_M_reading == true` && `_M_writing == true` is unused.

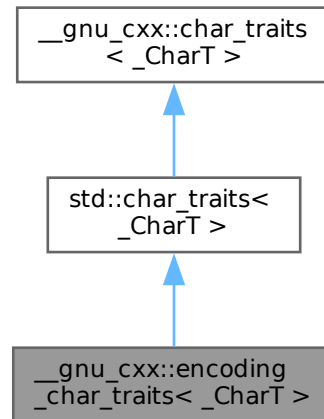
The documentation for this class was generated from the following file:

- [enc_filebuf.h](#)

6.426 `__gnu_cxx::encoding_char_traits<_CharT>` Struct Template Reference

```
#include <codecvt_specializations.h>
```

Inheritance diagram for `__gnu_cxx::encoding_char_traits<_CharT>`:



Public Types

- typedef `_CharT` **char_type**
- typedef `_Char_types<_CharT>::int_type` **int_type**
- typedef `_Char_types<_CharT>::off_type` **off_type**
- typedef `std::fpos<state_type>` **pos_type**
- typedef `encoding_state` **state_type**

Static Public Member Functions

- static constexpr void **assign** (char_type &__c1, const char_type &__c2)
- static constexpr char_type * **assign** (char_type *__s, std::size_t __n, char_type __a)
- static constexpr int **compare** (const char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr char_type * **copy** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2)
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static constexpr const char_type * **find** (const char_type *__s, std::size_t __n, const char_type &__a)
- static constexpr std::size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2)
- static constexpr char_type * **move** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c)
- static constexpr char_type **to_char_type** (const int_type &__c)
- static constexpr int_type **to_int_type** (const char_type &__c)

6.426.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::encoding_char_traits< _CharT >
```

`encoding_char_traits`

The documentation for this struct was generated from the following file:

- [codecvt_specializations.h](#)

6.427 `__gnu_cxx::encoding_state` Class Reference

```
#include <codecvt_specializations.h>
```

Public Types

- typedef `iconv_t` **descriptor_type**

Public Member Functions

- **encoding_state** (const char * __int, const char * __ext, int __ibom=0, int __ebom=0, int __bytes=1)
- **encoding_state** (const [encoding_state](#) & __obj)
- int **character_ratio** () const
- int **external_bom** () const
- const `std::string` **external_encoding** () const
- bool **good** () const throw ()
- const `descriptor_type` & **in_descriptor** () const
- int **internal_bom** () const
- const `std::string` **internal_encoding** () const
- [encoding_state](#) & **operator=** (const [encoding_state](#) & __obj)
- const `descriptor_type` & **out_descriptor** () const

Protected Member Functions

- void **construct** (const [encoding_state](#) & __obj)
- void **destroy** () throw ()
- void **init** ()

Protected Attributes

- int **_M_bytes**
- int **_M_ext_bom**
- `std::string` **_M_ext_enc**
- `descriptor_type` **_M_in_desc**
- int **_M_int_bom**
- `std::string` **_M_int_enc**
- `descriptor_type` **_M_out_desc**

6.427.1 Detailed Description

Extension to use `iconv` for dealing with character encodings.

The documentation for this class was generated from the following file:

- [codecvt_specializations.h](#)

6.428 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, No_Throw >` Struct Template Reference

6.428.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc, bool No_Throw>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, No_Throw >
```

Entry compare, primary template.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

6.429 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false >` Struct Template Reference

```
#include <entry_cmp.hpp>
```

Classes

- struct [type](#)

Public Types

- typedef `__rebind_v::const_pointer` `entry`

6.429.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false >
```

Specialization, false.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

6.430 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true >` Struct Template Reference

```
#include <entry_cmp.hpp>
```

Public Types

- typedef `Cmp_Fn` [type](#)

6.430.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true >
```

Specialization, true.

6.430.2 Member Typedef Documentation

`type`

```
template<typename _VTp , typename Cmp_Fn , typename _Alloc >
typedef Cmp_Fn \_\_gnu\_pbds::detail::entry\_cmp<_VTp, Cmp_Fn, _Alloc, true >::type
```

Compare.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

6.431 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, No_Throw>` Struct Template Reference

6.431.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc, bool No_Throw>
struct __gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, No_Throw>
```

Entry predicate primary class template.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

6.432 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false>` Struct Template Reference

```
#include <entry_pred.hpp>
```

Public Types

- typedef `__rebind_v::const_pointer` `entry`

6.432.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc>
struct __gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false>
```

Specialization, false.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

6.433 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, true>` Struct Template Reference

```
#include <entry_pred.hpp>
```

Public Types

- typedef `Pred` `type`

6.433.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc>
struct __gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, true>
```

Specialization, true.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

6.434 `__gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >` Struct Template Reference

`#include <eq_by_less.hpp>`
Inherits `Cmp_Fn`.

Public Member Functions

- `bool operator() (const Key &r_lhs, const Key &r_rhs) const`

6.434.1 Detailed Description

`template<typename Key, class Cmp_Fn>`
`struct __gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >`

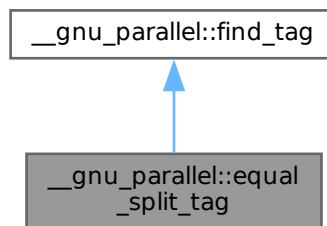
Equivalence function.

The documentation for this struct was generated from the following file:

- [eq_by_less.hpp](#)

6.435 `__gnu_parallel::equal_split_tag` Struct Reference

`#include <tags.h>`
Inheritance diagram for `__gnu_parallel::equal_split_tag`:



6.435.1 Detailed Description

Selects the equal splitting variant for `std::find()`.

See also

`_GLIBCXX_FIND_EQUAL_SPLIT`

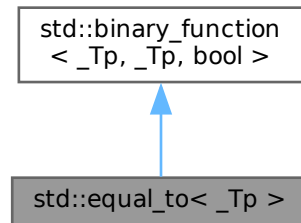
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.436 `std::equal_to< _Tp >` Struct Template Reference

`#include <stl_function.h>`

Inheritance diagram for std::equal_to< _Tp >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- constexpr bool **operator()**(const _Tp &__x, const _Tp &__y) const

6.436.1 Detailed Description

```
template<typename _Tp>
struct std::equal_to< _Tp >
```

One of the [comparison functors](#).

6.436.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary\_function< _Tp , _Tp , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.437 std::ranges::equal_to Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using **is_transparent** = __is_transparent

Public Member Functions

- template<typename _Tp, typename _Up >
requires equality_comparable_with<_Tp, _Up>
constexpr bool **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::declval<_Tp >()==std::declval<_Up >()))

6.437.1 Detailed Description

ranges::equal_to function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.438 std::equal_to< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp >(__t)==std::forward<_Up >(__u))) -> decltype(std::forward<_Tp >(__t)==std::forward<_Up >(__u))

6.438.1 Detailed Description

One of the [comparison functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.439 std::_V2::error_category Class Reference

```
#include <system_error>
```

Public Member Functions

- **error_category** (const [error_category](#) &)=delete
- virtual [error_condition](#) **default_error_condition** (int __i) const noexcept
- virtual bool **equivalent** (const [error_code](#) &__code, int __i) const noexcept
- virtual bool **equivalent** (int __i, const [error_condition](#) &__cond) const noexcept
- virtual [string](#) **message** (int) const =0
- virtual const char * **name** () const noexcept=0
- bool **operator!=** (const [error_category](#) &__other) const noexcept

- bool `operator<` (const `error_category` &__other) const noexcept
- `error_category` & `operator=` (const `error_category` &)=delete
- bool `operator==` (const `error_category` &__other) const noexcept

6.439.1 Detailed Description

Abstract base class for types defining a category of error codes.

An error category defines a context that gives meaning to the integer stored in an `error_code` or `error_condition` object. For example, the standard `errno` constants such as `EINVAL` and `ENOMEM` are associated with the "generic" category and other OS-specific error numbers are associated with the "system" category, but a user-defined category might give different meanings to the same numerical values.

A user-defined category can override the `equivalent` member functions to define correspondence between errors in different categories. For example, a category for errors from disk I/O could consider some of its error numbers equivalent to `ENOSPC` and `ENOENT` in the generic category.

Since

C++11

6.439.2 Member Function Documentation

`default_error_condition()`

```
virtual error_condition std::_V2::error_category::default_error_condition (
    int __i ) const [virtual], [noexcept]
```

Return an `error_condition` corresponding to `i` in this category.

`equivalent()` [1/2]

```
virtual bool std::_V2::error_category::equivalent (
    const error_code & __code,
    int __i ) const [virtual], [noexcept]
```

Test whether `code` corresponds to `i` for this category.

`equivalent()` [2/2]

```
virtual bool std::_V2::error_category::equivalent (
    int __i,
    const error_condition & __cond ) const [virtual], [noexcept]
```

Test whether `cond` corresponds to `i` for this category.

`name()`

```
virtual const char * std::_V2::error_category::name ( ) const [pure virtual], [noexcept]
```

A string that identifies the error category.

`operator<()`

```
bool std::_V2::error_category::operator< (
    const error_category & __other ) const [inline], [noexcept]
```

Ordered comparison that defines a total order for error categories.

`operator==()`

```
bool std::_V2::error_category::operator== (
    const error_category & __other ) const [inline], [noexcept]
```

An `error_category` only compares equal to itself.

The documentation for this class was generated from the following file:

- [system_error](#)

6.440 std::error_code Class Reference

```
#include <system_error>
```

Public Member Functions

- `template<typename _ErrorCodeEnum, typename = typename enable_if<is_error_code_enum<_ErrorCodeEnum>::value>::type> error_code (_ErrorCodeEnum __e) noexcept`
- `error_code (int __v, const error_category &__cat) noexcept`
- `void assign (int __v, const error_category &__cat) noexcept`
- `const error_category & category () const noexcept`
- `void clear () noexcept`
- `error_condition default_error_condition () const noexcept`
- `string message () const`
- `operator bool () const noexcept`
- `template<typename _ErrorCodeEnum > enable_if< is_error_code_enum< _ErrorCodeEnum >::value, error_code & >::type operator= (_ErrorCodeEnum __e) noexcept`
- `int value () const noexcept`

Related Symbols

(Note that these are not member symbols.)

- `error_code make_error_code (errc __e) noexcept`
- `bool operator!= (const error_code &__lhs, const error_code &__rhs) noexcept`
- `bool operator!= (const error_code &__lhs, const error_condition &__rhs) noexcept`
- `bool operator< (const error_code &__lhs, const error_code &__rhs) noexcept`
- `template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const error_code &__e)`

6.440.1 Detailed Description

Class `error_code`

This class is a value type storing an integer error number and a category that gives meaning to the error number. Typically this is done close to the point where the error happens, to capture the original error value.

An `error_code` object can be used to store the original error value emitted by some subsystem, with a category relevant to the subsystem. For example, errors from POSIX library functions can be represented by an `errno` value and the "generic" category, but errors from an HTTP library might be represented by an HTTP response status code (e.g. 404) and a custom category defined by the library.

Since

C++11

6.440.2 Member Function Documentation

`category()`

```
const error_category & std::error_code::category () const [inline], [noexcept]
```

The error category that this error belongs to.

default_error_condition()

```
error_condition std::error_code::default_error_condition ( ) const [noexcept]
```

An `error_condition` for this error's category and value.

message()

```
string std::error_code::message ( ) const [inline]
```

The category's description of the value.

operator bool()

```
std::error_code::operator bool ( ) const [inline], [explicit], [noexcept]
```

Test whether `value()` is non-zero.

value()

```
int std::error_code::value ( ) const [inline], [noexcept]
```

The error value.

The documentation for this class was generated from the following file:

- [system_error](#)

6.441 std::error_condition Class Reference

```
#include <system_error>
```

Public Member Functions

- [error_condition](#) () noexcept
- `template<typename _ErrorConditionEnum , typename = typename enable_if<is_error_condition_enum<_ErrorConditionEnum><↵
::value>::type>`
error_condition (_ErrorConditionEnum __e) noexcept
- [error_condition](#) (int __v, const [error_category](#) &__cat) noexcept
- `void` [assign](#) (int __v, const [error_category](#) &__cat) noexcept
- `const` [error_category](#) & [category](#) () const noexcept
- `void` [clear](#) () noexcept
- `string` [message](#) () const
- `operator bool` () const noexcept
- `template<typename _ErrorConditionEnum >`
`enable_if< is_error_condition_enum<_ErrorConditionEnum >::value, error_condition & >::type` **operator=** (_↵
_ErrorConditionEnum __e) noexcept
- `int` [value](#) () const noexcept

Related Symbols

(Note that these are not member symbols.)

- [error_condition](#) [make_error_condition](#) (errc __e) noexcept
- `bool` **operator!=** (const [error_condition](#) &__lhs, const [error_code](#) &__rhs) noexcept
- `bool` **operator!=** (const [error_condition](#) &__lhs, const [error_condition](#) &__rhs) noexcept
- `bool` **operator<** (const [error_condition](#) &__lhs, const [error_condition](#) &__rhs) noexcept
- `bool` **operator==** (const [error_code](#) &__lhs, const [error_code](#) &__rhs) noexcept
- `bool` **operator==** (const [error_code](#) &__lhs, const [error_condition](#) &__rhs) noexcept
- `bool` **operator==** (const [error_condition](#) &__lhs, const [error_code](#) &__rhs) noexcept
- `bool` **operator==** (const [error_condition](#) &__lhs, const [error_condition](#) &__rhs) noexcept

6.441.1 Detailed Description

Class `error_condition`

This class represents error conditions that may be visible at an API boundary. Different `error_code` values that can occur within a library or module might map to the same `error_condition`.

An `error_condition` represents something that the program can test for, and subsequently take appropriate action.

Since

C++11

6.441.2 Constructor & Destructor Documentation

`error_condition()` [1/2]

```
std::error_condition::error_condition ( ) [inline], [noexcept]
```

Initialize with a zero (no error) value and the generic category.

`error_condition()` [2/2]

```
std::error_condition::error_condition (
    int __v,
    const error_category & __cat ) [inline], [noexcept]
```

Initialize with the specified value and category.

6.441.3 Member Function Documentation

`assign()`

```
void std::error_condition::assign (
    int __v,
    const error_category & __cat ) [inline], [noexcept]
```

Set the value and category.

`category()`

```
const error_category & std::error_condition::category ( ) const [inline], [noexcept]
```

The error category that this error belongs to.

`clear()`

```
void std::error_condition::clear ( ) [inline], [noexcept]
```

Reset the value and category to the default-constructed state.

`message()`

```
string std::error_condition::message ( ) const [inline]
```

The category's description of the value.

`operator bool()`

```
std::error_condition::operator bool ( ) const [inline], [explicit], [noexcept]
```

Test whether value() is non-zero.

value()

```
int std::error_condition::value ( ) const [inline], [noexcept]
```

The error value.

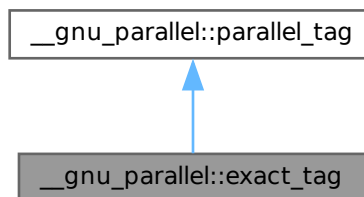
The documentation for this class was generated from the following file:

- [system_error](#)

6.442 `__gnu_parallel::exact_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::exact_tag`:

**Public Member Functions**

- **exact_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **__get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

6.442.1 Detailed Description

Forces parallel merging with exact splitting, at compile time.

6.442.2 Member Function Documentation**`__get_num_threads()`**

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

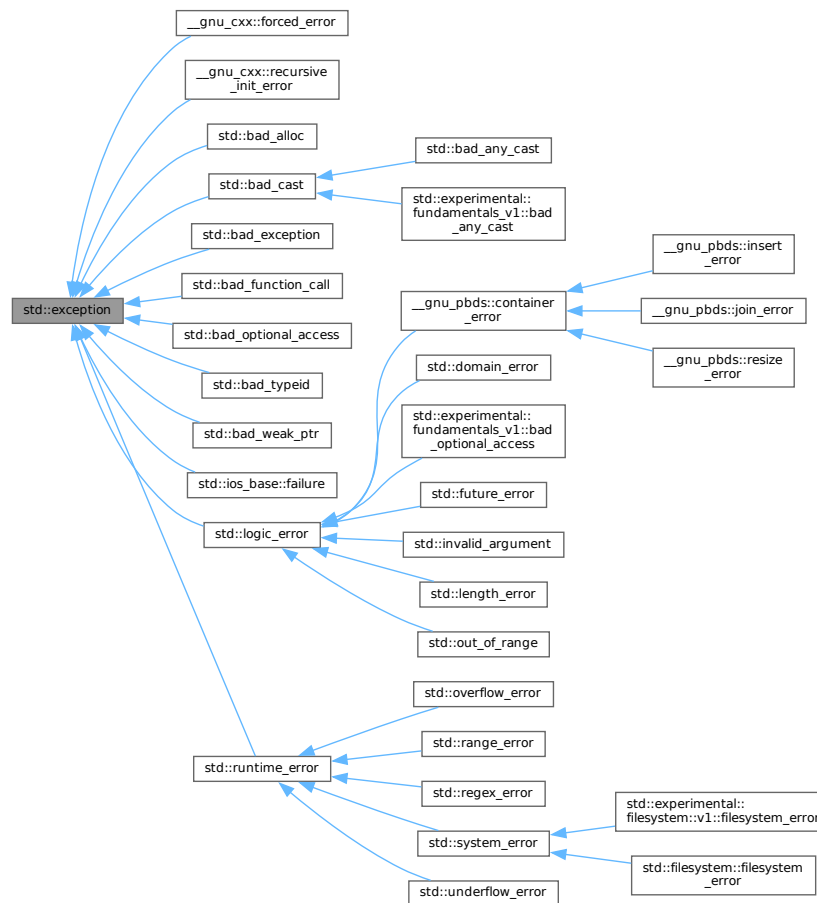
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.443 std::exception Class Reference

```
#include <exception.h>
```

Inheritance diagram for `std::exception`:



Public Member Functions

- **exception** (const [exception](#) &)=default
- **exception** ([exception](#) &&)=default
- [exception](#) & **operator=** (const [exception](#) &)=default
- [exception](#) & **operator=** ([exception](#) &&)=default
- virtual const char * [what](#) () const noexcept

6.443.1 Detailed Description

Base class for all library exceptions.

This is the base class for all exceptions thrown by the standard library, and by certain language expressions. You are free to derive your own exception classes, or use a different hierarchy, or to throw non-class data (e.g., fundamental types).

6.443.2 Member Function Documentation

what()

```
virtual const char * std::exception::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented in [std::bad_alloc](#), [std::ios_base::failure](#), [std::bad_exception](#), [std::bad_cast](#), [std::bad_typeid](#), [std::bad_any_cast](#), [std::filesystem::filesystem_error](#), [std::bad_weak_ptr](#), [std::bad_function_call](#), [std::experimental::fundamentals_v1::bad_weak_ptr](#), [std::experimental::filesystem_v1::filesystem_error](#), [std::future_error](#), [std::logic_error](#), [std::runtime_error](#), and [std::bad_optional_access](#).

The documentation for this class was generated from the following file:

- [exception.h](#)

6.444 std::__exception_ptr::exception_ptr Class Reference

```
#include <exception_ptr.h>
```

Public Member Functions

- **exception_ptr** (const [exception_ptr](#) &) noexcept
- **exception_ptr** ([exception_ptr](#) &&__o) noexcept
- **exception_ptr** (nullptr_t) noexcept
- const class [std::type_info](#) * **__cxa_exception_type** () const noexcept
- **operator bool** () const noexcept
- [exception_ptr](#) & **operator=** (const [exception_ptr](#) &) noexcept
- [exception_ptr](#) & **operator=** ([exception_ptr](#) &&__o) noexcept
- void **swap** ([exception_ptr](#) &) noexcept

Friends

- bool **operator==** (const [exception_ptr](#) &, const [exception_ptr](#) &) noexcept=default
- [exception_ptr](#) **std::current_exception** () noexcept
- template<typename _Ex >
[exception_ptr](#) **std::make_exception_ptr** (_Ex) noexcept
- void **std::rethrow_exception** ([exception_ptr](#))

Related Symbols

(Note that these are not member symbols.)

- void **swap** ([exception_ptr](#) &__lhs, [exception_ptr](#) &__rhs)

6.444.1 Detailed Description

An opaque pointer to an arbitrary exception.

The documentation for this class was generated from the following file:

- [exception_ptr.h](#)

6.445 `std::exponential_distribution<_RealType>` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- [exponential_distribution](#) ()
- [exponential_distribution](#) (`_RealType` __lambda)
- [exponential_distribution](#) (const [param_type](#) &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void [__generate](#) (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void [__generate](#) (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- template<typename `_UniformRandomNumberGenerator` >
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- `_RealType` [lambda](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) [operator\(\)](#) (`_UniformRandomNumberGenerator` &__urng)
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) [operator\(\)](#) (`_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- bool [operator==](#) (const [exponential_distribution](#) &__d1, const [exponential_distribution](#) &__d2)

6.445.1 Detailed Description

```
template<typename _RealType = double>
class std::exponential_distribution<_RealType>
```

An exponential continuous distribution for random numbers.

The formula for the exponential probability density function is $p(x|\lambda) = \lambda e^{-\lambda x}$.

Table 2079 Distribution Statistics

Mean	$\frac{1}{\lambda}$
Median	$\frac{\ln 2}{\lambda}$
Mode	<i>zero</i>
Range	$[0, \infty]$
Standard Deviation	$\frac{1}{\lambda}$

6.445.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::exponential_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.445.3 Constructor & Destructor Documentation

exponential_distribution() [1/2]

```
template<typename _RealType = double>
std::exponential_distribution< _RealType >::exponential_distribution ( ) [inline]
```

Constructs an exponential distribution with inverse scale parameter 1.0.

exponential_distribution() [2/2]

```
template<typename _RealType = double>
std::exponential_distribution< _RealType >::exponential_distribution (
    _RealType __lambda ) [inline], [explicit]
```

Constructs an exponential distribution with inverse scale parameter λ .

6.445.4 Member Function Documentation

lambda()

```
template<typename _RealType = double>
_RealType std::exponential_distribution< _RealType >::lambda ( ) const [inline]
```

Returns the inverse scale parameter of the distribution.

max()

```
template<typename _RealType = double>
result_type std::exponential_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits<_Tp>::max\(\)](#).

min()

```
template<typename _RealType = double>
result_type std::exponential_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::exponential_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::exponential_distribution<_RealType>::operator>\(\)](#).

Referenced by [std::exponential_distribution<_RealType>::operator>\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::exponential_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.
Referenced by [std::operator>>\(\)](#).

param() [2/2]

```
template<typename _RealType = double>
void std::exponential_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::exponential_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.
Has no effect on exponential distributions.

6.445.5 Friends And Related Symbol Documentation**operator==**

```
template<typename _RealType = double>
bool operator== (
    const exponential_distribution< _RealType > & __d1,
    const exponential_distribution< _RealType > & __d2 ) [friend]
```

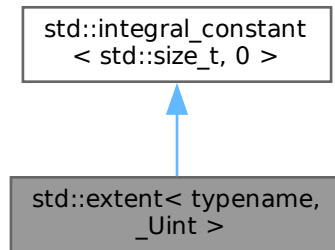
Return true if two exponential distributions have the same parameters.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.446 `std::extent< typename, _Uint >` Struct Template Reference

Inheritance diagram for `std::extent< typename, _Uint >`:

**Public Types**

- typedef [integral_constant](#)< `std::size_t`, `__v` > **type**
- typedef `std::size_t` **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `std::size_t` **value**

6.446.1 Detailed Description

```
template<typename, unsigned _Uint>
struct std::extent< typename, _Uint >
```

extent

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.447 `std::extreme_value_distribution< _RealType >` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- **extreme_value_distribution** (`_RealType __a, _RealType __b=_RealType(1)`)
- **extreme_value_distribution** (const `param_type` &__p)
- template<typename `_ForwardIterator`, typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng`)
- template<typename `_ForwardIterator`, typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- template<typename `_UniformRandomNumberGenerator` >
void **generate** (`result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `_RealType a` () const
- `_RealType b` () const
- `result_type max` () const
- `result_type min` () const
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng`)
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `param_type param` () const
- void `param` (const `param_type` &__param)
- void `reset` ()

Friends

- bool `operator==` (const `extreme_value_distribution` &__d1, const `extreme_value_distribution` &__d2)

6.447.1 Detailed Description

```
template<typename _RealType = double>
class std::extreme_value_distribution< _RealType >
```

A `extreme_value_distribution` random number distribution.
The formula for the normal probability mass function is

$$p(x|a, b) = \frac{1}{b} \exp\left(\frac{a-x}{b} - \exp\left(\frac{a-x}{b}\right)\right)$$

6.447.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::extreme_value_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.447.3 Member Function Documentation

a()

```
template<typename _RealType = double>
_RealType std::extreme_value_distribution< _RealType >::a ( ) const [inline]
```

Return the *a* parameter of the distribution.

b()

```
template<typename _RealType = double>
_RealType std::extreme_value_distribution<_RealType>::b ( ) const [inline]
```

Return the *b* parameter of the distribution.

max()

```
template<typename _RealType = double>
result_type std::extreme_value_distribution<_RealType>::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References `std::numeric_limits<_Tp>::max()`.

min()

```
template<typename _RealType = double>
result_type std::extreme_value_distribution<_RealType>::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.
References `std::numeric_limits<_Tp>::lowest()`.

operator()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::extreme_value_distribution<_RealType>::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References `std::extreme_value_distribution<_RealType>::operator()()`.

Referenced by `std::extreme_value_distribution<_RealType>::operator()()`.

param() [1/2]

```
template<typename _RealType = double>
param_type std::extreme_value_distribution<_RealType>::param ( ) const [inline]
```

Returns the parameter set of the distribution.
Referenced by `std::operator>>()`.

param() [2/2]

```
template<typename _RealType = double>
void std::extreme_value_distribution<_RealType>::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::extreme_value_distribution<_RealType>::reset ( ) [inline]
```

Resets the distribution state.

6.447.4 Friends And Related Symbol Documentation

operator==

```
template<typename _RealType = double>
bool operator== (
    const extreme\_value\_distribution< _RealType > & __d1,
    const extreme\_value\_distribution< _RealType > & __d2 ) [friend]
```

Return true if two extreme value distributions have the same parameters.

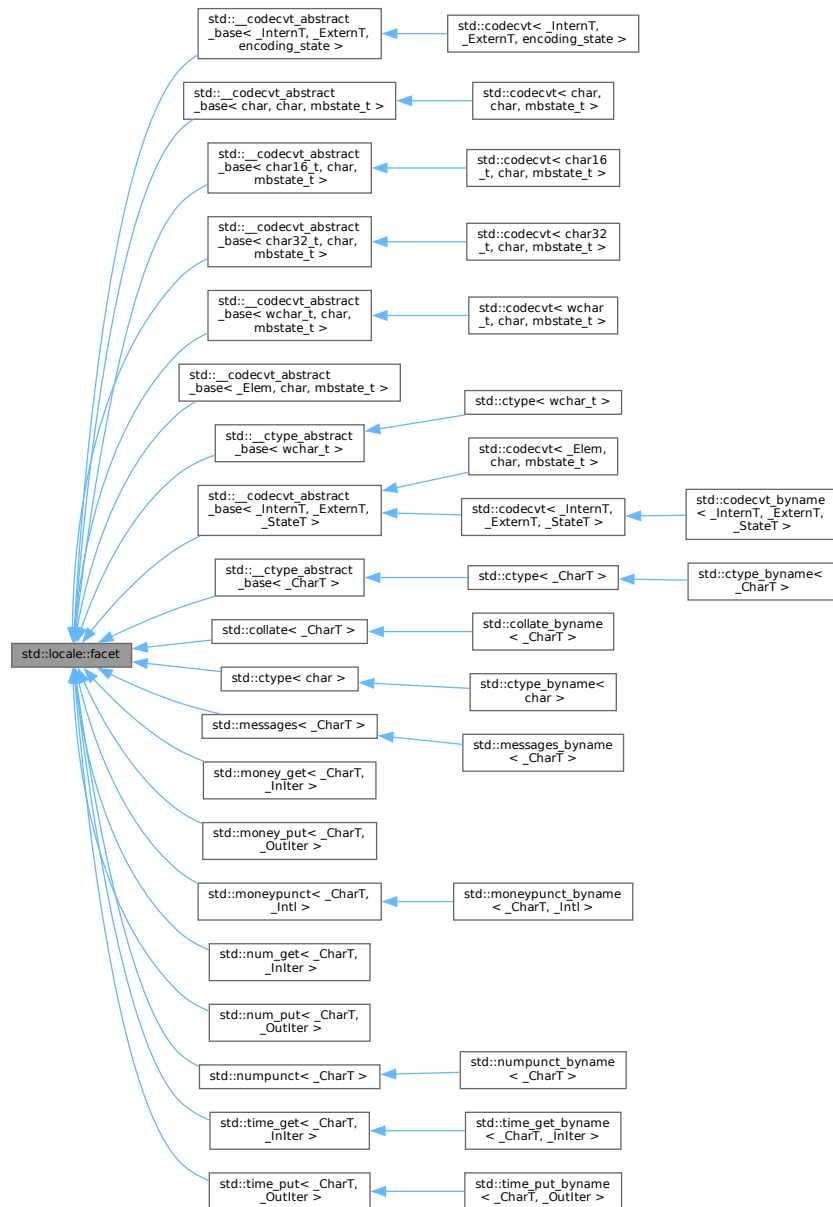
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.448 **std::locale::facet** Class Reference

```
#include <locale_classes.h>
```

Inheritance diagram for std::locale::facet:



Protected Member Functions

- **facet** (const **facet** &)=delete
- **facet** (size_t __refs=0) throw ()
- virtual ~**facet** ()
- **facet** & **operator=** (const **facet** &)=delete

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()

- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Friends

- class **locale**
- class **locale::_Impl**

6.448.1 Detailed Description

Localization functionality base class.

The facet class is the base class for a localization feature, such as money, time, and number printing. It provides common support for facets and reference management.

Facets may not be copied or assigned.

6.448.2 Constructor & Destructor Documentation

facet()

```
std::locale::facet::facet (
    size_t __refs = 0 ) throw ( )    [inline], [explicit], [protected]
```

Facet constructor.

This is the constructor provided by the standard. If refs is 0, the facet is destroyed when the last referencing locale is destroyed. Otherwise the facet will never be destroyed.

Parameters

<code>__refs</code>	The initial value for reference count.
---------------------	--

~facet()

```
virtual std::locale::facet::~~facet ( )    [protected], [virtual]
```

Facet destructor.

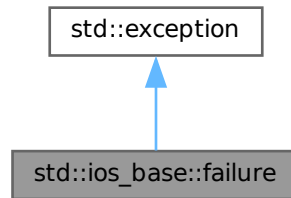
The documentation for this class was generated from the following file:

- [locale_classes.h](#)

6.449 std::ios_base::failure Class Reference

```
#include <ios_base.h>
```

Inheritance diagram for std::ios_base::failure:



Public Member Functions

- **failure** (const char * __s, const [error_code](#) &=error_code{})
- **failure** (const [string](#) & __s, const [error_code](#) &) noexcept
- **failure** (const [string](#) & __str) throw ()
- [error_code](#) **code** () const noexcept
- virtual const char * [what](#) () const throw ()

6.449.1 Detailed Description

These are thrown to indicate problems with io.

27.4.2.1.1 Class ios_base::failure

6.449.2 Member Function Documentation

what()

```
virtual const char * std::ios_base::failure::what ( ) const throw ( ) [virtual]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [ios_base.h](#)

6.450 std::filesystem::file_status Class Reference

```
#include <fs_dir.h>
```

Public Member Functions

- **file_status** (const [file_status](#) &) noexcept=default
- **file_status** ([file_status](#) &&) noexcept=default
- **file_status** ([file_type](#) __ft, [perms](#) __prms=perms::unknown) noexcept
- [file_status](#) & **operator=** (const [file_status](#) &) noexcept=default
- [file_status](#) & **operator=** ([file_status](#) &&) noexcept=default
- [perms](#) **permissions** () const noexcept
- void **permissions** ([perms](#) __prms) noexcept

- `file_type type ()` const noexcept
- `void type (file_type __ft)` noexcept

6.450.1 Detailed Description

Information about a file's type and permissions.

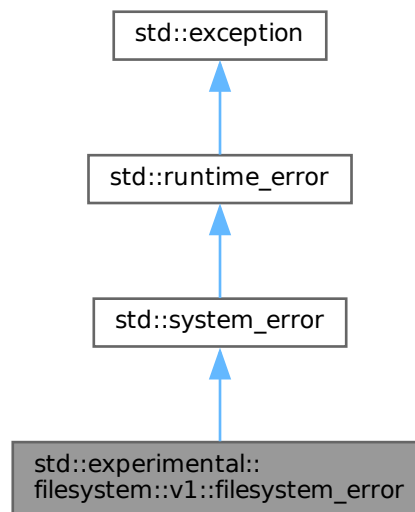
The documentation for this class was generated from the following file:

- [bits/fs_dir.h](#)

6.451 std::experimental::filesystem::v1::filesystem_error Class Reference

```
#include <fs_path.h>
```

Inheritance diagram for `std::experimental::filesystem::v1::filesystem_error`:



Public Member Functions

- `filesystem_error (const string &__what_arg, const path &__p1, const path &__p2, error_code __ec)`
- `filesystem_error (const string &__what_arg, const path &__p1, error_code __ec)`
- `filesystem_error (const string &__what_arg, error_code __ec)`
- `const error_code & code ()` const noexcept
- `const path & path1 ()` const noexcept
- `const path & path2 ()` const noexcept
- `const char * what ()` const noexcept

6.451.1 Detailed Description

Exception type thrown by the Filesystem TS library.

6.451.2 Member Function Documentation

what()

```
const char * std::experimental::filesystem::v1::filesystem_error::what ( ) const [inline], [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::runtime_error](#).

References [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#).

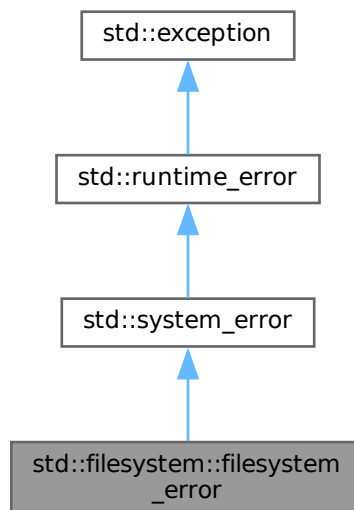
The documentation for this class was generated from the following file:

- [experimental/bits/fs_path.h](#)

6.452 std::filesystem::filesystem_error Class Reference

```
#include <fs_path.h>
```

Inheritance diagram for std::filesystem::filesystem_error:



Public Member Functions

- **filesystem_error** (const [filesystem_error](#) &)=default
- **filesystem_error** (const [string](#) &__what_arg, const [path](#) &__p1, const [path](#) &__p2, [error_code](#) __ec)
- **filesystem_error** (const [string](#) &__what_arg, const [path](#) &__p1, [error_code](#) __ec)
- **filesystem_error** (const [string](#) &__what_arg, [error_code](#) __ec)
- const [error_code](#) & **code** () const noexcept
- **filesystem_error** & **operator=** (const [filesystem_error](#) &)=default
- const [path](#) & **path1** () const noexcept
- const [path](#) & **path2** () const noexcept
- const char * **what** () const noexcept

6.452.1 Detailed Description

Exception type thrown by the Filesystem library.

6.452.2 Member Function Documentation

what()

```
const char * std::filesystem::filesystem_error::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::runtime_error](#).

References [std::basic_string<_CharT, _Traits, _Alloc>::data\(\)](#), and [std::make_error_code\(\)](#).

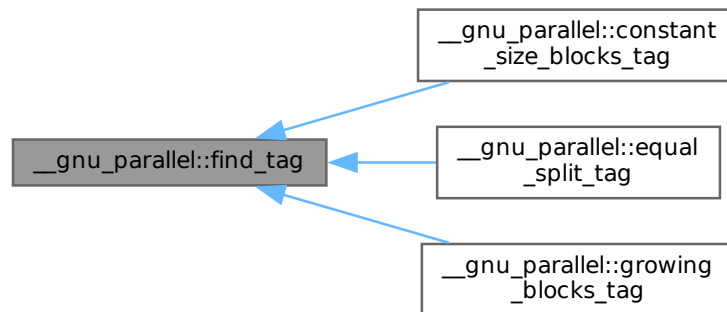
The documentation for this class was generated from the following file:

- [bits/fs_path.h](#)

6.453 __gnu_parallel::find_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::find_tag:



6.453.1 Detailed Description

Base class for for `std::find()` variants.

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.454 std::fisher_f_distribution<_RealType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **fisher_f_distribution** (_RealType __m, _RealType __n=_RealType(1))
- **fisher_f_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **m** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- _RealType **n** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::fisher_f_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [fisher_f_distribution](#) &__d1, const [fisher_f_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::fisher_f_distribution](#)< _RealType1 > &__x)

6.454.1 Detailed Description

template<typename _RealType = double>
class std::fisher_f_distribution<_RealType>

A fisher_f_distribution random number distribution.

The formula for the normal probability mass function is

$$p(x|m, n) = \frac{\Gamma((m+n)/2)}{\Gamma(m/2)\Gamma(n/2)} \left(\frac{m}{n}\right)^{m/2} x^{(m/2)-1} \left(1 + \frac{mx}{n}\right)^{-(m+n)/2}$$

6.454.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::fisher_f_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.454.3 Member Function Documentation

max()

```
template<typename _RealType = double>
result_type std::fisher_f_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits< _Tp >::max\(\)](#).

min()

```
template<typename _RealType = double>
result_type std::fisher_f_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::fisher_f_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

param() [1/2]

```
template<typename _RealType = double>
param_type std::fisher_f_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::fisher_f_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::fisher_f_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.
References [std::gamma_distribution< _RealType >::reset\(\)](#).

6.454.4 Friends And Related Symbol Documentation

operator<<

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::fisher_f_distribution< _RealType1 > & __x ) [friend]
```

Inserts a fisher_f_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A fisher_f_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const fisher_f_distribution< _RealType > & __d1,
    const fisher_f_distribution< _RealType > & __d2 ) [friend]
```

Return true if two Fisher f distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::fisher_f_distribution< _RealType1 > & __x ) [friend]
```

Extracts a fisher_f_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A fisher_f_distribution random number generator engine.

Returns

The input stream with __x extracted or in an error state.

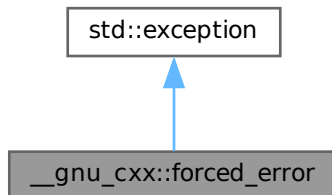
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.455 `__gnu_cxx::forced_error` Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for `__gnu_cxx::forced_error`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.455.1 Detailed Description

Thrown by utilities for testing exception safety.

6.455.2 Member Function Documentation

`what()`

```
virtual const char * std::exception::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented in [std::bad_alloc](#), [std::ios_base::failure](#), [std::bad_exception](#), [std::bad_cast](#), [std::bad_typeid](#), [std::bad_any_cast](#), [std::filesystem::filesystem_error](#), [std::bad_weak_ptr](#), [std::bad_function_call](#), [std::experimental::fundamentals_v1::bad](#), [std::experimental::filesystem::v1::filesystem_error](#), [std::future_error](#), [std::logic_error](#), [std::runtime_error](#), and [std::bad_optional_access](#).

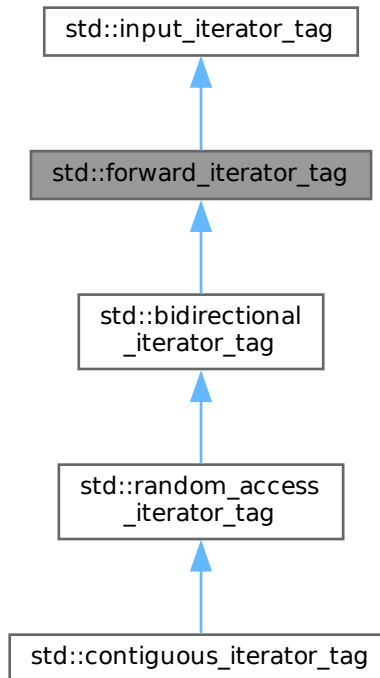
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.456 `std::forward_iterator_tag` Struct Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for `std::forward_iterator_tag`:



6.456.1 Detailed Description

Forward iterators support a superset of input iterator operations.
The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.457 `std::__debug::forward_list<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::__debug::forward_list<_Tp, _Alloc>`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::Safe_iterator<_Base_const_iterator, forward_list>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator<_Base_iterator, forward_list>` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `_Base::size_type` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **forward_list** (`_Base_ref __x`)
- template<typename `_InputIterator` , typename = `std::RequireInputIter<_InputIterator>>`
forward_list (`_InputIterator __first, _InputIterator __last, const allocator_type &__al=allocator_type()`)
- **forward_list** (`const allocator_type &__al`) noexcept
- **forward_list** (`const forward_list &`)=default
- **forward_list** (`const forward_list &__list, const allocator_type &__al`)
- **forward_list** (`forward_list &&`)=default
- **forward_list** (`forward_list &&__list, const allocator_type &__al`) noexcept(`std::is_nothrow_constructible<_Base, _Base, const allocator_type &>::value`)
- **forward_list** (`size_type __n, const __type_identity_t<_Tp> &__value, const allocator_type &__al=allocator_type()`)
- **forward_list** (`size_type __n, const allocator_type &__al=allocator_type()`)
- **forward_list** (`std::initializer_list<_Tp> __il, const allocator_type &__al=allocator_type()`)
- `const _Base &_M_base` () const noexcept
- `_Base &_M_base` () noexcept
- template<typename `_InputIterator` , typename = `std::RequireInputIter<_InputIterator>>`
void **assign** (`_InputIterator __first, _InputIterator __last`)
- void **assign** (`size_type __n, const _Tp &__val`)
- void **assign** (`std::initializer_list<_Tp> __il`)
- `const_iterator before_begin` () const noexcept
- `iterator before_begin` () noexcept
- `const_iterator begin` () const noexcept
- `iterator begin` () noexcept
- `const_iterator cbefore_begin` () const noexcept
- `const_iterator cbegin` () const noexcept
- `const_iterator cend` () const noexcept
- void **clear** () noexcept
- template<typename... `_Args`>
`iterator emplace_after` (`const_iterator __pos, _Args &&... __args`)
- `const_iterator end` () const noexcept
- `iterator end` () noexcept
- `iterator erase_after` (`const_iterator __pos`)
- `iterator erase_after` (`const_iterator __pos, const_iterator __last`)
- reference **front** ()
- `const_reference front` () const
- template<typename `_InputIterator` , typename = `std::RequireInputIter<_InputIterator>>`
`iterator insert_after` (`const_iterator __pos, _InputIterator __first, _InputIterator __last`)

- [iterator insert_after](#) ([const_iterator](#) __pos, [_Tp](#) && __val)
- [iterator insert_after](#) ([const_iterator](#) __pos, [const _Tp](#) & __val)
- [iterator insert_after](#) ([const_iterator](#) __pos, [size_type](#) __n, [const _Tp](#) & __val)
- [iterator insert_after](#) ([const_iterator](#) __pos, [std::initializer_list](#)< [_Tp](#) > __il)
- void **merge** ([forward_list](#) && __list)
- [template](#)<[typename](#) _Comp >
void **merge** ([forward_list](#) && __list, _Comp __comp)
- void **merge** ([forward_list](#) & __list)
- [template](#)<[typename](#) _Comp >
void **merge** ([forward_list](#) & __list, _Comp __comp)
- [forward_list](#) & **operator=** ([const forward_list](#) &)=default
- [forward_list](#) & **operator=** ([forward_list](#) &&)=default
- [forward_list](#) & **operator=** ([std::initializer_list](#)< [_Tp](#) > __il)
- void **pop_front** ()
- [__remove_return_type](#) **remove** ([const _Tp](#) & __val)
- [template](#)<[typename](#) _Pred >
[__remove_return_type](#) **remove_if** (_Pred __pred)
- void **resize** ([size_type](#) __sz)
- void **resize** ([size_type](#) __sz, [const value_type](#) & __val)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) && __list)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) && __list, [const_iterator](#) __before, [const_iterator](#) __last)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) && __list, [const_iterator](#) __i)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) & __list)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) & __list, [const_iterator](#) __before, [const_iterator](#) __last)
- void **splice_after** ([const_iterator](#) __pos, [forward_list](#) & __list, [const_iterator](#) __i)
- void **swap** ([forward_list](#) & __list) noexcept(noexcept([declval](#)< [_Base](#) & >().swap(__list)))
- [__remove_return_type](#) **unique** ()
- [template](#)<[typename](#) _BinPred >
[__remove_return_type](#) **unique** (_BinPred __binary_pred)

Protected Member Functions

- void **_M_swap** (_Safe_container & __x) noexcept

Friends

- [template](#)<[typename](#) _ItT, [typename](#) _SeqT, [typename](#) _CatT >
class ::[__gnu_debug::Safe_iterator](#)

6.457.1 Detailed Description

[template](#)<[typename](#) _Tp, [typename](#) _Alloc = [std::allocator](#)<_Tp>>>
class [std::__debug::forward_list](#)< _Tp, _Alloc >

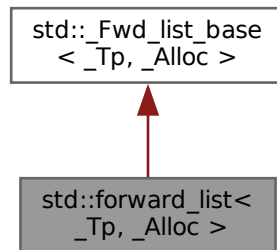
Class [std::forward_list](#) with safety/checking/debug instrumentation.
The documentation for this class was generated from the following file:

- [debug/forward_list](#)

6.458 std::forward_list<_Tp, _Alloc> Class Template Reference

```
#include <forward_list.h>
```

Inheritance diagram for std::forward_list<_Tp, _Alloc>:



Public Types

- typedef `_Alloc allocator_type`
- typedef `_Base::const_iterator const_iterator`
- typedef `_Alloc_traits::const_pointer const_pointer`
- typedef `const value_type & const_reference`
- typedef `std::ptrdiff_t difference_type`
- typedef `_Base::iterator iterator`
- typedef `_Alloc_traits::pointer pointer`
- typedef `value_type & reference`
- typedef `std::size_t size_type`
- typedef `_Tp value_type`

Public Member Functions

- `forward_list()`=default
- `template<typename InputIterator, typename = std::_RequireInputIter<InputIterator>>>`
`forward_list (InputIterator __first, InputIterator __last, const _Alloc &__al=_Alloc())`
- `forward_list (const _Alloc &__al) noexcept`
- `forward_list (const forward_list &__list)`
- `forward_list (const forward_list &__list, const __type_identity_t<_Alloc> &__al)`
- `forward_list (forward_list &&)=default`
- `forward_list (forward_list &&__list, const __type_identity_t<_Alloc> &__al) noexcept` (`_Node_alloc_traits::S` ← `_always_equal()`)
- `forward_list (size_type __n, const _Alloc &__al=_Alloc())`
- `forward_list (size_type __n, const _Tp &__value, const _Alloc &__al=_Alloc())`
- `forward_list (std::initializer_list<_Tp> __il, const _Alloc &__al=_Alloc())`
- `~forward_list () noexcept`
- `template<typename InputIterator, typename = std::_RequireInputIter<InputIterator>>>`
`void assign (InputIterator __first, InputIterator __last)`
- `void assign (size_type __n, const _Tp &__val)`
- `void assign (std::initializer_list<_Tp> __il)`

- [const_iterator before_begin](#) () const noexcept
- [iterator before_begin](#) () noexcept
- [const_iterator begin](#) () const noexcept
- [iterator begin](#) () noexcept
- [const_iterator cbefore_begin](#) () const noexcept
- [const_iterator cbegin](#) () const noexcept
- [const_iterator cend](#) () const noexcept
- void [clear](#) () noexcept
- template<typename... _Args>
 [iterator emplace_after](#) (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
 reference [emplace_front](#) (_Args &&... __args)
- bool [empty](#) () const noexcept
- [const_iterator end](#) () const noexcept
- [iterator end](#) () noexcept
- [iterator erase_after](#) (const_iterator __pos)
- [iterator erase_after](#) (const_iterator __pos, const_iterator __last)
- reference [front](#) ()
- const_reference [front](#) () const
- allocator_type [get_allocator](#) () const noexcept
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>
 [iterator insert_after](#) (const_iterator __pos, _InputIterator __first, _InputIterator __last)
- [iterator insert_after](#) (const_iterator __pos, _Tp &&__val)
- [iterator insert_after](#) (const_iterator __pos, const _Tp &__val)
- [iterator insert_after](#) (const_iterator __pos, size_type __n, const _Tp &__val)
- [iterator insert_after](#) (const_iterator __pos, std::initializer_list<_Tp> __il)
- size_type [max_size](#) () const noexcept
- void [merge](#) (forward_list &&__list)
- template<typename _Comp>
 void [merge](#) (forward_list &&__list, _Comp __comp)
- void **merge** (forward_list &__list)
- template<typename _Comp>
 void **merge** (forward_list &__list, _Comp __comp)
- forward_list & operator= (const forward_list &__list)
- forward_list & operator= (forward_list &&__list) noexcept(_Node_alloc_traits::_S_nothrow_move())
- forward_list & operator= (std::initializer_list<_Tp> __il)
- void [pop_front](#) ()
- void **push_front** (_Tp &&__val)
- void [push_front](#) (const _Tp &__val)
- __remove_return_type [remove](#) (const _Tp &__val)
- template<typename _Pred>
 __remove_return_type [remove_if](#) (_Pred __pred)
- template<typename _Pred>
 auto **remove_if** (_Pred __pred) -> __remove_return_type
- void [resize](#) (size_type __sz)
- void [resize](#) (size_type __sz, const value_type &__val)
- void [reverse](#) () noexcept
- void [sort](#) ()
- template<typename _Comp>
 void [sort](#) (_Comp __comp)
- void [splice_after](#) (const_iterator __pos, forward_list &&__list) noexcept

- void `splice_after` (`const_iterator` __pos, `forward_list` && __list, `const_iterator` __i) noexcept
- void `splice_after` (`const_iterator` __pos, `forward_list` & __list) noexcept
- void `splice_after` (`const_iterator` __pos, `forward_list` & __list, `const_iterator` __i) noexcept
- void `swap` (`forward_list` & __list) noexcept
- `__remove_return_type` `unique` ()
- template<typename `_BinPred` >
 `__remove_return_type` `unique` (`_BinPred` __binary_pred)
- template<typename `_BinPred` >
 auto `unique` (`_BinPred` __binary_pred) -> `__remove_return_type`
- void `splice_after` (`const_iterator` __pos, `forward_list` &&, `const_iterator` __before, `const_iterator` __last) noexcept
- void `splice_after` (`const_iterator` __pos, `forward_list` &, `const_iterator` __before, `const_iterator` __last) noexcept

6.458.1 Detailed Description

```
template<typename _Tp, typename _Alloc = allocator<_Tp>>
class std::forward_list< _Tp, _Alloc >
```

A standard container with linear time access to elements, and fixed time insertion/deletion at any point in the sequence.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [sequence](#), including the [optional sequence requirements](#) with the exception of `at` and `operator[]`.

This is a *singly linked* list. Traversal up the list requires linear time, but adding and removing elements (or *nodes*) is done in constant time, regardless of where the change takes place. Unlike `std::vector` and `std::deque`, random-access iterators are not provided, so subscripting (`[]`) access is not allowed. For algorithms which only need sequential access, this lack makes no difference.

Also unlike the other standard containers, `std::forward_list` provides specialized algorithms unique to linked lists, such as splicing, sorting, and in-place reversal.

6.458.2 Constructor & Destructor Documentation

`forward_list()` [1/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list ( ) [default]
```

Creates a `forward_list` with no elements.

`forward_list()` [2/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    const _Alloc & __al ) [inline], [explicit], [noexcept]
```

Creates a `forward_list` with no elements.

Parameters

<code>__al</code>	An allocator object.
-------------------	----------------------

forward_list() [3/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    const forward_list< _Tp, _Alloc > & __list,
    const __type_identity_t< _Alloc > & __al ) [inline]
```

Copy constructor with allocator argument.

Parameters

<code>__list</code>	Input list to copy.
<code>__al</code>	An allocator object.

References [std::forward_list<_Tp, _Alloc >::begin\(\)](#), and [std::forward_list<_Tp, _Alloc >::end\(\)](#).

forward_list() [4/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    forward_list< _Tp, _Alloc > && __list,
    const __type_identity_t< _Alloc > & __al ) [inline], [noexcept]
```

Move constructor with allocator argument.

Parameters

<code>__list</code>	Input list to move.
<code>__al</code>	An allocator object.

forward_list() [5/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    size_type __n,
    const _Alloc & __al = _Alloc() ) [inline], [explicit]
```

Creates a forward_list with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__al</code>	An allocator object.

This constructor creates the forward_list with `__n` default constructed elements.

forward_list() [6/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    size_type __n,
    const _Tp & __value,
    const _Alloc & __al = _Alloc() ) [inline]
```

Creates a forward_list with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__al</code>	An allocator object.

This constructor fills the `forward_list` with `__n` copies of `__value`.

forward_list() [7/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
std::forward_list< _Tp, _Alloc >::forward_list (
    _InputIterator __first,
    _InputIterator __last,
    const _Alloc & __al = _Alloc() ) [inline]
```

Builds a `forward_list` from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__al</code>	An allocator object.

Create a `forward_list` consisting of copies of the elements from `[__first,__last)`. This is linear in N (where N is distance(`↔` `__first,__last`)).

forward_list() [8/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    const forward_list< _Tp, _Alloc > & __list ) [inline]
```

The `forward_list` copy constructor.

Parameters

<code>__list</code>	A <code>forward_list</code> of identical element and allocator types.
---------------------	---

References `std::forward_list< _Tp, _Alloc >::begin()`, and `std::forward_list< _Tp, _Alloc >::end()`.

forward_list() [9/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    forward_list< _Tp, _Alloc > && ) [default]
```

The `forward_list` move constructor.

Parameters

<code>__list</code>	A <code>forward_list</code> of identical element and allocator types.
---------------------	---

The newly-created `forward_list` contains the exact contents of the moved instance. The contents of the moved instance are a valid, but unspecified `forward_list`.

forward_list() [10/10]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::forward_list (
    std::initializer_list< _Tp > __il,
    const _Alloc & __al = _Alloc() ) [inline]
```

Builds a forward_list from an initializer_list.

Parameters

<code>__il</code>	An initializer_list of value_type.
<code>__al</code>	An allocator object.

Create a forward_list consisting of copies of the elements in the initializer_list `__il`. This is linear in `__il.size()`.

~forward_list()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
std::forward_list< _Tp, _Alloc >::~~forward_list ( ) [inline], [noexcept]
```

The forward_list dtor.

6.458.3 Member Function Documentation**assign()** [1/3]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
void std::forward_list< _Tp, _Alloc >::assign (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Assigns a range to a forward_list.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a forward_list with copies of the elements in the range `[__first, __last)`.

Note that the assignment completely changes the forward_list and that the number of elements of the resulting forward_list is the same as the number of elements assigned.

Referenced by `std::forward_list< _Tp, _Alloc >::assign()`, and `std::forward_list< _Tp, _Alloc >::operator=()`.

assign() [2/3]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list< _Tp, _Alloc >::assign (
    size_type __n,
    const _Tp & __val ) [inline]
```

Assigns a given value to a forward_list.

Parameters

<code>__n</code>	Number of elements to be assigned.
------------------	------------------------------------

Parameters

<code>__val</code>	Value to be assigned.
--------------------	-----------------------

This function fills a `forward_list` with `__n` copies of the given value. Note that the assignment completely changes the `forward_list`, and that the resulting `forward_list` has `__n` elements.

assign() [3/3]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list< _Tp, _Alloc >::assign (
    std::initializer_list< _Tp > __il ) [inline]
```

Assigns an `initializer_list` to a `forward_list`.

Parameters

<code>__il</code>	An <code>initializer_list</code> of <code>value_type</code> .
-------------------	---

Replace the contents of the `forward_list` with copies of the elements in the `initializer_list` `__il`. This is linear in `il.size()`. References [std::forward_list< _Tp, _Alloc >::assign\(\)](#).

before_begin() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
const_iterator std::forward_list< _Tp, _Alloc >::before_begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points before the first element in the `forward_list`. Iteration is done in ordinary element order.

before_begin() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list< _Tp, _Alloc >::before_begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points before the first element in the `forward_list`. Iteration is done in ordinary element order.

Referenced by [std::forward_list< _Tp, _Alloc >::insert_after\(\)](#), and [std::forward_list< _Tp, _Alloc >::insert_after\(\)](#).

begin() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
const_iterator std::forward_list< _Tp, _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `forward_list`. Iteration is done in ordinary element order.

begin() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list< _Tp, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the `forward_list`. Iteration is done in ordinary element order.

Referenced by [std::forward_list< _Tp, _Alloc >::forward_list\(\)](#), and [std::forward_list< _Tp, _Alloc >::forward_list\(\)](#).

cbefore_begin()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
```

```
const_iterator std::forward_list< _Tp, _Alloc >::cbefore_begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points before the first element in the forward_list. Iteration is done in ordinary element order.

Referenced by `std::forward_list< _Tp, _Alloc >::emplace_front()`, `std::forward_list< _Tp, _Alloc >::push_front()`, and `std::forward_list< _Tp, _Alloc >::remove()`.

cbegin()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
```

```
const_iterator std::forward_list< _Tp, _Alloc >::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the forward_list. Iteration is done in ordinary element order.

Referenced by `std::operator<()`, `std::forward_list< _Tp, _Alloc >::operator=()`, and `std::operator==(())`.

cend()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
```

```
const_iterator std::forward_list< _Tp, _Alloc >::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the forward_list. Iteration is done in ordinary element order.

Referenced by `std::operator<()`, `std::forward_list< _Tp, _Alloc >::operator=()`, and `std::operator==(())`.

clear()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
```

```
void std::forward_list< _Tp, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all the elements.

Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

emplace_after()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
```

```
template<typename... _Args>
```

```
iterator std::forward_list< _Tp, _Alloc >::emplace_after (
```

```
    const_iterator __pos,
```

```
    _Args &&... __args ) [inline]
```

Constructs object in forward_list after the specified iterator.

Parameters

<code>__pos</code>	A const_iterator into the forward_list.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type T constructed with `T(std::forward<Args>(args)...) after the specified location. Due to the nature of a forward_list this operation can be done in constant time, and does not invalidate iterators and references.`

emplace_front()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
template<typename... _Args>
reference std::forward_list< _Tp, _Alloc >::emplace_front (
    _Args &&... __args ) [inline]
```

Constructs object in forward_list at the front of the list.

Parameters

<code>__args</code>	Arguments.
---------------------	------------

This function will insert an object of type `Tp` constructed with `Tp(std::forward<Args>(args)...)` at the front of the list. Due to the nature of a `forward_list` this operation can be done in constant time, and does not invalidate iterators and references.

References [std::forward_list< _Tp, _Alloc >::cbefore_begin\(\)](#), and [std::forward_list< _Tp, _Alloc >::front\(\)](#).

empty()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
bool std::forward_list< _Tp, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the `forward_list` is empty. (Thus `begin()` would equal `end()`.)

Referenced by [std::forward_list< _Tp, _Alloc >::insert_after\(\)](#).

end() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
const_iterator std::forward_list< _Tp, _Alloc >::end ( ) const [inline], [noexcept]
```

Returns a read-only iterator that points one past the last element in the `forward_list`. Iteration is done in ordinary element order.

end() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list< _Tp, _Alloc >::end ( ) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the `forward_list`. Iteration is done in ordinary element order.

Referenced by [std::forward_list< _Tp, _Alloc >::forward_list\(\)](#), [std::forward_list< _Tp, _Alloc >::forward_list\(\)](#), [std::forward_list< _Tp, _Alloc >::insert_after\(\)](#), and [std::forward_list< _Tp, _Alloc >::insert_after\(\)](#).

erase_after() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list< _Tp, _Alloc >::erase_after (
    const_iterator __pos ) [inline]
```

Removes the element pointed to by the iterator following `pos`.

Parameters

<code>__pos</code>	Iterator pointing before element to be erased.
--------------------	--

Returns

An iterator pointing to the element following the one that was erased, or end() if no such element exists.

This function will erase the element at the given position and thus shorten the forward_list by one.

Due to the nature of a forward_list this operation can be done in constant time, and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase_after() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list< _Tp, _Alloc >::erase_after (
    const_iterator __pos,
    const_iterator __last ) [inline]
```

Remove a range of elements.

Parameters

<code>__pos</code>	Iterator pointing before the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

@ `__last`.

This function will erase the elements in the range (`__pos`, `__last`) and shorten the forward_list accordingly.

This operation is linear time in the size of the range and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

front() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
reference std::forward_list< _Tp, _Alloc >::front ( ) [inline]
```

Returns a read/write reference to the data at the first element of the forward_list.
Referenced by `std::forward_list< _Tp, _Alloc >::emplace_front()`.

front() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
const_reference std::forward_list< _Tp, _Alloc >::front ( ) const [inline]
```

Returns a read-only (constant) reference to the data at the first element of the forward_list.

get_allocator()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
allocator_type std::forward_list< _Tp, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

insert_after() [1/4]

```
template<typename _Tp , typename _Alloc >
template<typename _InputIterator , typename >
forward_list< _Tp, _Alloc >::iterator forward_list::insert_after (
    const_iterator __pos,
```



```

    __InputIterator __first,
    __InputIterator __last )

```

Inserts a range into the `forward_list`.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator pointing to the last inserted element or `__pos` if `__first == __last`.

This function will insert copies of the data in the range `[__first,__last)` into the `forward_list` after the location specified by `__pos`.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

References [std::forward_list<_Tp, _Alloc>::before_begin\(\)](#), [std::forward_list<_Tp, _Alloc>::empty\(\)](#), and [std::forward_list<_Tp, _Alloc>::size\(\)](#).

`insert_after()` [2/4]

```

template<typename _Tp , typename _Alloc = allocator<_Tp>>
iterator std::forward_list<_Tp, _Alloc>::insert_after (
    const_iterator __pos,
    const _Tp & __val ) [inline]

```

Inserts given value into `forward_list` after specified iterator.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__val</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value after the specified location. Due to the nature of a `forward_list` this operation can be done in constant time, and does not invalidate iterators and references.

Referenced by [std::forward_list<_Tp, _Alloc>::insert_after\(\)](#).

`insert_after()` [3/4]

```

template<typename _Tp , typename _Alloc >
forward_list<_Tp, _Alloc>::iterator forward_list::insert_after (
    const_iterator __pos,
    size_type __n,
    const _Tp & __val )

```

Inserts a number of copies of given data into the `forward_list`.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__n</code>	Number of elements to be inserted.
<code>__val</code>	Data to be inserted.

Returns

An iterator pointing to the last inserted copy of *val* or *pos* if *n* == 0.

This function will insert a specified number of copies of the given data after the location specified by *pos*. This operation is linear in the number of elements inserted and does not invalidate iterators and references. References `std::forward_list<_Tp, _Alloc>::before_begin()`, and `std::forward_list<_Tp, _Alloc>::end()`.

insert_after() [4/4]

```
template<typename _Tp, typename _Alloc = allocator<_Tp>>
iterator std::forward_list<_Tp, _Alloc>::insert_after (
    const_iterator __pos,
    std::initializer_list<_Tp> __il ) [inline]
```

Inserts the contents of an `initializer_list` into `forward_list` after the specified iterator.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__il</code>	An <code>initializer_list</code> of <code>value_type</code> .

Returns

An iterator pointing to the last inserted element or `__pos` if `__il` is empty.

This function will insert copies of the data in the `initializer_list` `__il` into the `forward_list` before the location specified by `__pos`.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

References `std::forward_list<_Tp, _Alloc>::insert_after()`.

max_size()

```
template<typename _Tp, typename _Alloc = allocator<_Tp>>
size_type std::forward_list<_Tp, _Alloc>::max_size ( ) const [inline], [noexcept]
```

Returns the largest possible number of elements of `forward_list`.

References `__gnu_cxx::__alloc_traits<_Alloc, typename>::max_size()`.

merge() [1/2]

```
template<typename _Tp, typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc>::merge (
    forward_list<_Tp, _Alloc> && __list ) [inline]
```

Merge sorted lists.

Parameters

<code>__list</code>	Sorted list to merge.
---------------------	-----------------------

Assumes that both *list* and this list are sorted according to `operator<()`. Merges elements of `__list` into this list in sorted order, leaving `__list` empty when complete. Elements in this list precede elements in `__list` that are equal.

References `std::forward_list<_Tp, _Alloc>::merge()`, and `std::move()`.

Referenced by `std::forward_list<_Tp, _Alloc>::merge()`.

merge() [2/2]

```
template<typename _Tp , typename _Alloc >
template<typename _Comp >
void forward_list::merge (
    forward_list< _Tp, _Alloc > && __list,
    _Comp __comp )
```

Merge sorted lists according to comparison function.

Parameters

<code>__list</code>	Sorted list to merge.
<code>__comp</code>	Comparison function defining sort order.

Assumes that both `__list` and this list are sorted according to `comp`. Merges elements of `__list` into this list in sorted order, leaving `__list` empty when complete. Elements in this list precede elements in `__list` that are equivalent according to `comp()`.

References [std::__addressof\(\)](#), and [std::move\(\)](#).

operator=() [1/3]

```
template<typename _Tp , typename _Alloc >
forward_list< _Tp, _Alloc > & forward_list::operator= (
    const forward_list< _Tp, _Alloc > & __list )
```

The `forward_list` assignment operator.

Parameters

<code>__list</code>	A <code>forward_list</code> of identical element and allocator types.
---------------------	---

All the elements of `__list` are copied.

Whether the allocator is copied depends on the allocator traits.

References [std::__addressof\(\)](#), [std::forward_list< _Tp, _Alloc >::cbegin\(\)](#), and [std::forward_list< _Tp, _Alloc >::cend\(\)](#).

operator=() [2/3]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
forward_list & std::forward_list< _Tp, _Alloc >::operator= (
    forward_list< _Tp, _Alloc > && __list ) [inline], [noexcept]
```

The `forward_list` move assignment operator.

Parameters

<code>__list</code>	A <code>forward_list</code> of identical element and allocator types.
---------------------	---

The contents of `__list` are moved into this `forward_list` (without copying, if the allocators permit it).

Afterwards `__list` is a valid, but unspecified `forward_list`

Whether the allocator is moved depends on the allocator traits.

References [std::move\(\)](#).

operator=() [3/3]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
forward_list & std::forward_list< _Tp, _Alloc >::operator= (
```

```
std::initializer_list<_Tp> __il ) [inline]
```

The `forward_list` initializer list assignment operator.

Parameters

<code>__il</code>	An <code>initializer_list</code> of <code>value_type</code> .
-------------------	---

Replace the contents of the `forward_list` with copies of the elements in the `initializer_list __il`. This is linear in `__il.size()`.
References [std::forward_list<_Tp, _Alloc >::assign\(\)](#).

`pop_front()`

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc >::pop_front ( ) [inline]
```

Removes first element.

This is a typical stack operation. It shrinks the `forward_list` by one. Due to the nature of a `forward_list` this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop_front()` is called.

`push_front()`

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc >::push_front (
    const _Tp & __val ) [inline]
```

Add data to the front of the `forward_list`.

Parameters

<code>__val</code>	Data to be added.
--------------------	-------------------

This is a typical stack operation. The function creates an element at the front of the `forward_list` and assigns the given data to it. Due to the nature of a `forward_list` this operation can be done in constant time, and does not invalidate iterators and references.

References [std::forward_list<_Tp, _Alloc >::cbefore_begin\(\)](#).

`remove()`

```
template<typename _Tp , typename _Alloc >
auto forward_list::remove (
    const _Tp & __val )
```

Remove all elements equal to value.

Parameters

<code>__val</code>	The value to remove.
--------------------	----------------------

Removes every element in the list equal to `__val`. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::forward_list<_Tp, _Alloc >::cbefore_begin\(\)](#), and [std::forward_list<_Tp, _Alloc >::splice_after\(\)](#).

remove_if()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
template<typename _Pred >
__remove_return_type std::forward_list< _Tp, _Alloc >::remove_if (
    _Pred __pred )
```

Remove all elements satisfying a predicate.

Parameters

<code>__pred</code>	Unary predicate function or object.
---------------------	-------------------------------------

Removes every element in the list for which the predicate returns true. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

resize() [1/2]

```
template<typename _Tp , typename _Alloc >
void forward_list::resize (
    size_type __sz )
```

Resizes the forward_list to the specified number of elements.

Parameters

<code>__sz</code>	Number of elements the forward_list should contain.
-------------------	---

This function will resize the forward_list to the specified number of elements. If the number is smaller than the forward_list's current number of elements the forward_list is truncated, otherwise the forward_list is extended and the new elements are default constructed.

resize() [2/2]

```
template<typename _Tp , typename _Alloc >
void forward_list::resize (
    size_type __sz,
    const value_type & __val )
```

Resizes the forward_list to the specified number of elements.

Parameters

<code>__sz</code>	Number of elements the forward_list should contain.
<code>__val</code>	Data with which new elements should be populated.

This function will resize the forward_list to the specified number of elements. If the number is smaller than the forward_list's current number of elements the forward_list is truncated, otherwise the forward_list is extended and new elements are populated with given data.

reverse()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list< _Tp, _Alloc >::reverse ( ) [inline], [noexcept]
```

Reverse the elements in list.

Reverse the order of elements in the list in linear time.

sort() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc >::sort ( ) [inline]
```

Sort the elements of the list.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

References `std::forward_list<_Tp, _Alloc >::sort()`.

Referenced by `std::forward_list<_Tp, _Alloc >::sort()`.

sort() [2/2]

```
template<typename _Tp , class _Alloc >
template<typename _Comp >
void forward_list::sort (
    _Comp __comp )
```

Sort the `forward_list` using a comparison function.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

splice_after() [1/4]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc >::splice_after (
    const_iterator __pos,
    forward_list<_Tp, _Alloc > && ,
    const_iterator __before,
    const_iterator __last ) [inline], [noexcept]
```

Insert range from another `forward_list`.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__before</code>	Iterator referencing before the start of range in list.
<code>__last</code>	Iterator referencing the end of range in list.

Removes elements in the range (`__before`, `__last`) and inserts them after `__pos` in constant time.

Undefined if `__pos` is in (`__before`, `__last`).

splice_after() [2/4]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list<_Tp, _Alloc >::splice_after (
    const_iterator __pos,
    forward_list<_Tp, _Alloc > && __list ) [inline], [noexcept]
```

Insert contents of another `forward_list`.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.

The elements of `list` are inserted in constant time after the element referenced by `pos`. `list` becomes an empty list.

Requires this != x.

Referenced by `std::forward_list<_Tp, _Alloc >::remove()`.

splice_after() [3/4]

```
template<typename _Tp , typename _Alloc >
void forward_list::splice_after (
    const_iterator __pos,
    forward_list< _Tp, _Alloc > && __list,
    const_iterator __i ) [noexcept]
```

Insert element from another forward_list.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__i</code>	Iterator referencing the element before the element to move.

Removes the element in list *list* referenced by *i* and inserts it into the current list after *pos*.

splice_after() [4/4]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list< _Tp, _Alloc >::splice_after (
    const_iterator __pos,
    forward_list< _Tp, _Alloc > & ,
    const_iterator __before,
    const_iterator __last ) [inline], [noexcept]
```

Insert range from another forward_list.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__before</code>	Iterator referencing before the start of range in list.
<code>__last</code>	Iterator referencing the end of range in list.

Removes elements in the range (`__before`,`__last`) and inserts them after `__pos` in constant time. Undefined if `__pos` is in (`__before`,`__last`).

swap()

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
void std::forward_list< _Tp, _Alloc >::swap (
    forward_list< _Tp, _Alloc > & __list ) [inline], [noexcept]
```

Swaps data with another forward_list.

Parameters

<code>__list</code>	A forward_list of the same element and allocator types.
---------------------	---

This exchanges the elements between two lists in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(l1,l2)` will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

References [std::swap\(\)](#).

unique() [1/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
__remove_return_type std::forward_list<_Tp, _Alloc>::unique ( ) [inline]
```

Remove consecutive duplicate elements.

For each consecutive set of elements with the same value, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References `std::forward_list<_Tp, _Alloc>::unique()`.

Referenced by `std::forward_list<_Tp, _Alloc>::unique()`.

unique() [2/2]

```
template<typename _Tp , typename _Alloc = allocator<_Tp>>
template<typename _BinPred >
__remove_return_type std::forward_list<_Tp, _Alloc>::unique (
    _BinPred __binary_pred )
```

Remove consecutive elements satisfying a predicate.

Parameters

<code>__binary_pred</code>	Binary predicate function or object.
----------------------------	--------------------------------------

For each consecutive set of elements [first,last) that satisfy predicate(first,i) where i is an iterator in [first,last), remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

The documentation for this class was generated from the following files:

- [forward_list.h](#)
- [forward_list.tcc](#)

6.459 `std::fpos<_StateT>` Class Template Reference

```
#include <postypes.h>
```

Public Member Functions

- **fpos** (const [fpos](#) &)=default
- [fpos](#) (streamoff __off)
- [operator streamoff](#) () const
- [fpos operator+](#) (streamoff __off) const
- [fpos & operator+=](#) (streamoff __off)
- [streamoff operator-](#) (const [fpos](#) &__other) const
- [fpos operator-](#) (streamoff __off) const
- [fpos & operator-=](#) (streamoff __off)
- [fpos & operator=](#) (const [fpos](#) &)=default
- [_StateT state](#) () const
- void [state](#) (_StateT __st)

6.459.1 Detailed Description

```
template<typename _StateT>
class std::fpos<_StateT>
```

Class representing stream positions.

The standard places no requirements upon the template parameter StateT. In this implementation StateT must be DefaultConstructible, CopyConstructible and Assignable. The standard only requires that fpos should contain a member of type StateT. In this implementation it also contains an offset stored as a signed integer.

Parameters

<i>StateT</i>	Type passed to and returned from state().
---------------	---

6.459.2 Constructor & Destructor Documentation

fpos()

```
template<typename _StateT >
std::fpos< _StateT >::fpos (
    streamoff __off ) [inline]
```

Construct position from offset.

6.459.3 Member Function Documentation

operator streamoff()

```
template<typename _StateT >
std::fpos< _StateT >::operator streamoff ( ) const [inline]
```

Convert to streamoff.

operator+()

```
template<typename _StateT >
fpos std::fpos< _StateT >::operator+ (
    streamoff __off ) const [inline]
```

Add position and offset.

operator+=()

```
template<typename _StateT >
fpos & std::fpos< _StateT >::operator+= (
    streamoff __off ) [inline]
```

Add offset to this position.

operator-() [1/2]

```
template<typename _StateT >
streamoff std::fpos< _StateT >::operator- (
    const fpos< _StateT > & __other ) const [inline]
```

Subtract position to return offset.

operator-() [2/2]

```
template<typename _StateT >
fpos std::fpos< _StateT >::operator- (
    streamoff __off ) const [inline]
```

Subtract offset from position.

operator-=()

```
template<typename _StateT >
fpos & std::fpos< _StateT >::operator-= (
    streamoff __off ) [inline]
```

Subtract offset from this position.

state() [1/2]

```
template<typename _StateT >
_StateT std::fpos< _StateT >::state ( ) const [inline]
```

Return the last set value of *st*.

state() [2/2]

```
template<typename _StateT >
void std::fpos< _StateT >::state (
    _StateT __st ) [inline]
```

Remember the value of *st*.

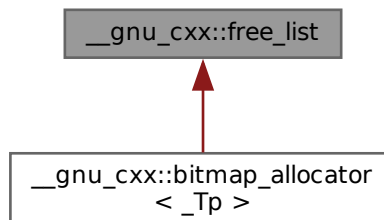
The documentation for this class was generated from the following file:

- [postypes.h](#)

6.460 __gnu_cxx::free_list Class Reference

```
#include <bitmap_allocator.h>
```

Inheritance diagram for __gnu_cxx::free_list:

**Public Types**

- typedef __mutex **mutex_type**
- typedef vector_type::iterator **iterator**
- typedef std::size_t * **value_type**
- typedef __detail::__mini_vector< value_type > **vector_type**

Public Member Functions

- void `_M_clear` ()
- std::size_t * `_M_get` (std::size_t __sz)
- void `_M_insert` (std::size_t * __addr) throw ()

6.460.1 Detailed Description

The free list class for managing chunks of memory to be given to and returned by the `bitmap_allocator`.

6.460.2 Member Function Documentation

`_M_clear()`

```
void __gnu_cxx::free_list::_M_clear ( )
```

This function just clears the internal Free List, and gives back all the memory to the OS.

`_M_get()`

```
std::size_t * __gnu_cxx::free_list::_M_get (
    std::size_t __sz )
```

This function gets a block of memory of the specified size from the free list.

Parameters

<code>__sz</code>	The size in bytes of the memory required.
-------------------	---

Returns

A pointer to the new memory block of size at least equal to that requested.

`_M_insert()`

```
void __gnu_cxx::free_list::_M_insert (
    std::size_t * __addr ) throw ( ) [inline]
```

This function returns the block of memory to the internal free list.

Parameters

<code>__addr</code>	The pointer to the memory block that was given by a call to the <code>_M_get</code> function.
---------------------	---

Referenced by `__gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object()`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

6.461 `std::from_chars_result` Struct Reference

Public Attributes

- `errc ec`
- `const char * ptr`

Friends

- `bool operator== (const from_chars_result &, const from_chars_result &)=default`

6.461.1 Detailed Description

Result type of `std::from_chars`.

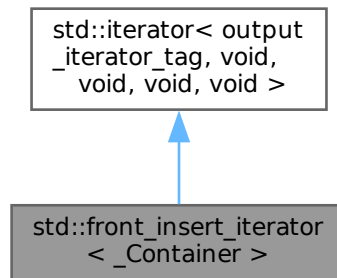
The documentation for this struct was generated from the following file:

- [charconv](#)

6.462 std::front_insert_iterator< _Container > Class Template Reference

```
#include <stl_iterator.h>
```

Inheritance diagram for std::front_insert_iterator< _Container >:



Public Types

- typedef _Container [container_type](#)
- using **difference_type** = ptrdiff_t
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

Public Member Functions

- constexpr [front_insert_iterator](#) (_Container &__x)
- constexpr [front_insert_iterator](#) & [operator*](#) ()
- constexpr [front_insert_iterator](#) & [operator++](#) ()
- constexpr [front_insert_iterator](#) [operator++](#) (int)
- constexpr [front_insert_iterator](#) & [operator=](#) (const typename _Container::value_type &__value)
- constexpr [front_insert_iterator](#) & [operator=](#) (typename _Container::value_type &&__value)

Protected Attributes

- _Container * **container**

6.462.1 Detailed Description

```
template<typename _Container>
class std::front_insert_iterator< _Container >
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator prepends it to the container using push_front.

Tip: Using the front_inserter function to create these iterators can save typing.

6.462.2 Member Typedef Documentation

container_type

```
template<typename _Container >
typedef _Container std::front_insert_iterator< _Container >::container_type
```

A nested typedef for the type of whatever container you used.

iterator_category

```
typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >←
::iterator_category [inherited]
```

One of the [tag types](#).

pointer

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer [inherited]
```

This type represents a pointer-to-value_type.

reference

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference [inherited]
```

This type represents a reference-to-value_type.

value_type

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type [inherited]
```

The type "pointed to" by the iterator.

6.462.3 Constructor & Destructor Documentation

front_insert_iterator()

```
template<typename _Container >
constexpr std::front_insert_iterator< _Container >::front_insert_iterator (
    _Container & __x ) [inline], [explicit], [constexpr]
```

The only way to create this iterator is with a container.

6.462.4 Member Function Documentation

operator*()

```
template<typename _Container >
constexpr front_insert_iterator & std::front_insert_iterator< _Container >::operator* ( ) [inline],
[constexpr]
```

Simply returns *this.

operator++() [1/2]

```
template<typename _Container >
constexpr front_insert_iterator & std::front_insert_iterator< _Container >::operator++ ( ) [inline],
[constexpr]
```

Simply returns *this. (This iterator does not *move*.)

operator++() [2/2]

```
template<typename _Container >
constexpr front_insert_iterator std::front_insert_iterator< _Container >::operator++ (
    int ) [inline], [constexpr]
```

Simply returns *this. (This iterator does not *move*.)

operator=()

```
template<typename _Container >
constexpr front_insert_iterator & std::front_insert_iterator< _Container >::operator= (
    const typename _Container::value_type & __value ) [inline], [constexpr]
```

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const T for <code>container<T></code> .
----------------------	---

Returns

This iterator, for chained operations.

This kind of iterator doesn't really have a *position* in the container (you can think of the position as being permanently at the front, if you like). Assigning a value to the iterator will always prepend the value to the front of the container.

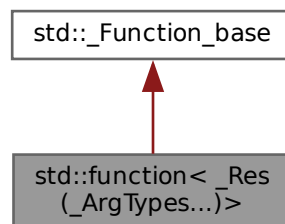
The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.463 std::function<_Res(_ArgTypes...)> Class Template Reference

```
#include <std_function.h>
```

Inheritance diagram for `std::function<_Res(_ArgTypes...)>`:

**Public Types**

- typedef `_Res` **result_type**

Public Member Functions

- [function](#) () noexcept

- `template<typename _Functor, typename _Constraints = _Requires<_Callable<_Functor>>>>
function (_Functor &&__f) noexcept(_Handler<_Functor>::template _S_nothrow_init<_Functor>())`
- `function (const function &__x)`
- `function (function &&__x) noexcept`
- `function (nullptr_t) noexcept`
- `operator bool () const noexcept`
- `_Res operator() (_ArgTypes... __args) const`
- `template<typename _Functor >
_Requires<_Callable<_Functor>, function & > operator= (_Functor &&__f) noexcept(_Handler<_Functor>
>::template _S_nothrow_init<_Functor>())`
- `function & operator= (const function &__x)`
- `function & operator= (function &&__x) noexcept`
- `function & operator= (nullptr_t) noexcept`
- `template<typename _Functor >
function & operator= (reference_wrapper<_Functor> __f) noexcept`
- `void swap (function &__x) noexcept`
- `const type_info & target_type () const noexcept`
- `template<typename _Functor >
_Functor * target () noexcept`
- `template<typename _Functor >
const _Functor * target () const noexcept`

6.463.1 Detailed Description

`template<typename _Res, typename... _ArgTypes>
class std::function<_Res(_ArgTypes...)>`

Polymorphic function wrapper.

Since

C++11

6.463.2 Constructor & Destructor Documentation

`function()` [1/5]

```
template<typename _Res, typename... _ArgTypes>
std::function<_Res(_ArgTypes...)>::function () [inline], [noexcept]
```

Default construct creates an empty function call wrapper.

Postcondition

`!(bool)*this`

`function()` [2/5]

```
template<typename _Res, typename... _ArgTypes>
std::function<_Res(_ArgTypes...)>::function (
    nullptr_t ) [inline], [noexcept]
```

Creates an empty function call wrapper.

Postcondition

`!(bool)*this`

function() [3/5]

```
template<typename _Res , typename... _ArgTypes>
std::function< _Res(_ArgTypes...)>::function (
    const function< _Res(_ArgTypes...)> & __x ) [inline]
```

Function copy constructor.

Parameters

<code>__x</code>	A function object with identical call signature.
------------------	--

Postcondition

```
bool(*this) == bool(__x)
```

The newly-created function contains a copy of the target of `__x` (if it has one).

function() [4/5]

```
template<typename _Res , typename... _ArgTypes>
std::function< _Res(_ArgTypes...)>::function (
    function< _Res(_ArgTypes...)> && __x ) [inline], [noexcept]
```

Function move constructor.

Parameters

<code>__x</code>	A function object rvalue with identical call signature.
------------------	---

The newly-created function contains the target of `__x` (if it has one).

function() [5/5]

```
template<typename _Res , typename... _ArgTypes>
template<typename _Functor , typename _Constraints = _Requires<_Callable<_Functor>>>
std::function< _Res(_ArgTypes...)>::function (
    _Functor && __f ) [inline], [noexcept]
```

Builds a function that targets a copy of the incoming function object.

Parameters

<code>__f</code>	A function object that is callable with parameters of type <code>ArgTypes...</code> and returns a value convertible to <code>Res</code> .
------------------	---

The newly-created function object will target a copy of `__f`. If `__f` is `reference_wrapper<F>`, then this function object will contain a reference to the function object `__f.get()`. If `__f` is a null function pointer, null pointer-to-member, or empty `std::function`, the newly-created object will be empty.

If `__f` is a non-null function pointer or an object of type `reference_wrapper<F>`, this function will not throw.

6.463.3 Member Function Documentation

operator bool()

```
template<typename _Res , typename... _ArgTypes>
std::function< _Res(_ArgTypes...)>::operator bool ( ) const [inline], [explicit], [noexcept]
```

Determine if the function wrapper has a target.

Returns

`true` when this function object contains a target, or `false` when it is empty.

This function will not throw exceptions.

operator()()

```
template<typename _Res , typename... _ArgTypes>
_Res std::function< _Res(_ArgTypes...)>::operator() (
    _ArgTypes... __args ) const [inline]
```

Invokes the function targeted by `*this`.

Returns

the result of the target.

Exceptions

<code>'bad_function_call'</code>	when <code>!(bool)*this</code>
----------------------------------	--------------------------------

The function call operator invokes the target function object stored by `this`.

operator=() [1/5]

```
template<typename _Res , typename... _ArgTypes>
template<typename _Functor >
_Requires< _Callable< _Functor >, function & > std::function< _Res(_ArgTypes...)>::operator= (
    _Functor && __f ) [inline], [noexcept]
```

Function assignment to a new target.

Parameters

<code>↔</code>	A function object that is callable with parameters of type <code>_ArgTypes . . .</code> and returns a value convertible to
<code>↔</code>	<code>_Res</code> .
<code>↔</code>	
<code>↔</code>	
<code>f</code>	

Returns

`*this`

Since

C++11

This function object wrapper will target a copy of `__f`. If `__f` is `reference_wrapper<F>`, then this function object will contain a reference to the function object `__f.get()`. If `__f` is a null function pointer or null pointer-to-member, this object will be empty.

If `__f` is a non-null function pointer or an object of type `reference_wrapper<F>`, this function will not throw.

operator=() [2/5]

```
template<typename _Res , typename... _ArgTypes>
function & std::function< _Res(_ArgTypes...)>::operator= (
    const function< _Res(_ArgTypes...)> & __x ) [inline]
```

Function assignment operator.

Parameters

<code>__x</code>	A function with identical call signature.
------------------	---

Postcondition

```
(bool)*this == (bool)x
```

Returns

```
*this
```

The target of `__x` is copied to `*this`. If `__x` has no target, then `*this` will be empty.

If `__x` targets a function pointer or a reference to a function object, then this operation will not throw an exception.

operator=() [3/5]

```
template<typename _Res , typename... _ArgTypes>
function & std::function< _Res(_ArgTypes...)>::operator= (
    function< _Res(_ArgTypes...)> && __x ) [inline], [noexcept]
```

Function move-assignment operator.

Parameters

<code>__x</code>	A function rvalue with identical call signature.
------------------	--

Returns

```
*this
```

The target of `__x` is moved to `*this`. If `__x` has no target, then `*this` will be empty.

If `__x` targets a function pointer or a reference to a function object, then this operation will not throw an exception.

References [std::move\(\)](#).

operator=() [4/5]

```
template<typename _Res , typename... _ArgTypes>
function & std::function< _Res(_ArgTypes...)>::operator= (
    nullptr_t ) [inline], [noexcept]
```

Function assignment to empty.

Postcondition

`! (bool) *this`

Returns

`*this`

The target of `*this` is deallocated, leaving it empty.

operator=() [5/5]

```
template<typename _Res , typename... _ArgTypes>
template<typename _Functor >
function & std::function< _Res(_ArgTypes...)>::operator= (
    reference_wrapper< _Functor > __f ) [inline], [noexcept]
```

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

swap()

```
template<typename _Res , typename... _ArgTypes>
void std::function< _Res(_ArgTypes...)>::swap (
    function< _Res(_ArgTypes...)> & __x ) [inline], [noexcept]
```

Swap the targets of two function objects.

Parameters

<code>__x</code>	A function with identical call signature.
------------------	---

Swap the targets of `this` function object and `__f`. This function will not throw exceptions.

References [std::swap\(\)](#).

target() [1/2]

```
template<typename _Res , typename... _ArgTypes>
template<typename _Functor >
const _Functor * std::function< _Res(_ArgTypes...)>::target ( ) const [inline], [noexcept]
```

Access the stored target function object.

Returns

Returns a pointer to the stored target function object, if `typeid(_Functor).equals(target_type())`; otherwise, a null pointer.

This function does not throw exceptions.

target() [2/2]

```
template<typename _Res , typename... _ArgTypes>
template<typename _Functor >
_Functor * std::function< _Res(_ArgTypes...)>::target ( ) [inline], [noexcept]
```

Access the stored target function object.

Returns

Returns a pointer to the stored target function object, if `typeid(_Functor).equals(target_type())`; otherwise, a null pointer.

This function does not throw exceptions.

target_type()

```
template<typename _Res , typename... _ArgTypes>
const type\_info & std::function< _Res(_ArgTypes...)>::target_type ( ) const [inline], [noexcept]
```

Determine the type of the target of this function object wrapper.

Returns

the type identifier of the target function object, or `typeid(void)` if `!(bool)*this`.

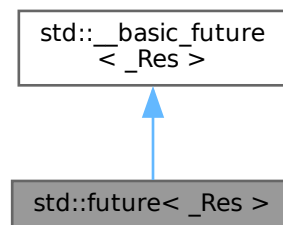
This function will not throw exceptions.

The documentation for this class was generated from the following file:

- [std_function.h](#)

6.464 `std::future<_Res>` Class Template Reference

Inheritance diagram for `std::future<_Res>`:

**Public Member Functions**

- **future** (const [future](#) &)=delete
- **future** ([future](#) &&__uf) noexcept
- **_Res** [get](#) ()
- **future** & **operator=** (const [future](#) &)=delete
- **future** & **operator=** ([future](#) &&__fut) noexcept
- **shared_future**<_Res> **share** () noexcept
- bool **valid** () const noexcept
- void **wait** () const
- template<typename _Rep , typename _Period >
future_status **wait_for** (const [chrono::duration](#)<_Rep, _Period> &__rel) const
- template<typename _Clock , typename _Duration >
future_status **wait_until** (const [chrono::time_point](#)<_Clock, _Duration> &__abs) const

Protected Types

- typedef `__future_base::Result<_Res>` & `__result_type`

Protected Member Functions

- `__result_type _M_get_result()` const
- void `_M_swap` (`__basic_future` &__that) noexcept

Friends

- template<typename `_Fn`, typename... `_Args`>
`future`< `__async_result_of<_Fn, _Args...>` > `async` (`launch`, `_Fn` &&, `_Args` &&...)
- template<typename >
class `packaged_task`
- class `promise`< `_Res` >

6.464.1 Detailed Description

template<typename `_Res`>
class `std::future`< `_Res` >

Primary template for future.

6.464.2 Constructor & Destructor Documentation

`future()`

```
template<typename _Res >  
std::future< _Res >::future (   
    future< _Res > && __uf ) [inline], [noexcept]
```

Move constructor.

6.464.3 Member Function Documentation

`_M_get_result()`

```
template<typename _Res >  
__result_type std::__basic_future< _Res >::_M_get_result ( ) const [inline], [protected], [inherited]
```

Wait for the state to be ready and rethrow any stored exception.

`get()`

```
template<typename _Res >  
_Res std::future< _Res >::get ( ) [inline]
```

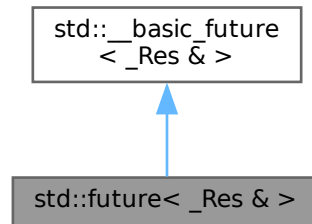
Retrieving the value.

The documentation for this class was generated from the following file:

- `future`

6.465 std::future<_Res &> Class Template Reference

Inheritance diagram for std::future<_Res &>:



Public Member Functions

- **future** (const [future](#) &)=delete
- **future** ([future](#) &&__uf) noexcept
- [_Res](#) & **get** ()
- **future** & **operator=** (const [future](#) &)=delete
- **future** & **operator=** ([future](#) &&__fut) noexcept
- [shared_future](#)<_Res &> **share** () noexcept
- bool **valid** () const noexcept
- void **wait** () const
- [future_status](#) **wait_for** (const [chrono::duration](#)<_Rep, _Period > &__rel) const
- [future_status](#) **wait_until** (const [chrono::time_point](#)<_Clock, _Duration > &__abs) const

Protected Types

- typedef __future_base::_Result<_Res &> & __**result_type**

Protected Member Functions

- __result_type **_M_get_result** () const
- void **_M_swap** (__[basic_future](#) &__that) noexcept

Friends

- template<typename _Fn, typename... _Args>
[future](#)<__async_result_of<_Fn, _Args... >> **async** ([launch](#), _Fn &&, _Args &&...)
- template<typename >
class **packaged_task**
- class **promise**<_Res &>

6.465.1 Detailed Description

template<typename _Res>
class std::future<_Res &>

Partial specialization for future<R&>

6.465.2 Constructor & Destructor Documentation

future()

```
template<typename _Res >
std::future< _Res & >::future (
    future< _Res & > && __uf ) [inline], [noexcept]
```

Move constructor.

6.465.3 Member Function Documentation

_M_get_result()

```
__result_type std::__basic_future< _Res & >::_M_get_result ( ) const [inline], [protected],
[inherited]
```

Wait for the state to be ready and rethrow any stored exception.

get()

```
template<typename _Res >
_Res & std::future< _Res & >::get ( ) [inline]
```

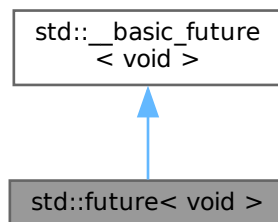
Retrieving the value.

The documentation for this class was generated from the following file:

- [future](#)

6.466 std::future< void > Class Reference

Inheritance diagram for std::future< void >:



Public Member Functions

- **future** (const [future](#) &)=delete
- **future** ([future](#) &&__uf) noexcept
- void **get** ()
- **future** & **operator=** (const [future](#) &)=delete
- **future** & **operator=** ([future](#) &&__fut) noexcept
- [shared_future](#)< void > **share** () noexcept
- bool **valid** () const noexcept
- void **wait** () const

- `future_status wait_for` (const `chrono::duration< _Rep, _Period > &__rel`) const
- `future_status wait_until` (const `chrono::time_point< _Clock, _Duration > &__abs`) const

Protected Types

- typedef `__future_base::Result< void > & __result_type`

Protected Member Functions

- `__result_type _M_get_result` () const
- `void _M_swap` (`__basic_future &__that`) noexcept

Friends

- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async` (`launch`, `_Fn &&`, `_Args &&...`)
- `template<typename >`
class `packaged_task`
- class `promise< void >`

6.466.1 Detailed Description

Explicit specialization for `future<void>`

6.466.2 Constructor & Destructor Documentation

`future()`

```
std::future< void >::future (
    future< void > && __uf ) [inline], [noexcept]
```

Move constructor.

6.466.3 Member Function Documentation

`_M_get_result()`

```
__result_type std::__basic_future< void >::_M_get_result ( ) const [inline], [protected], [inherited]
```

Wait for the state to be ready and rethrow any stored exception.

`get()`

```
void std::future< void >::get ( ) [inline]
```

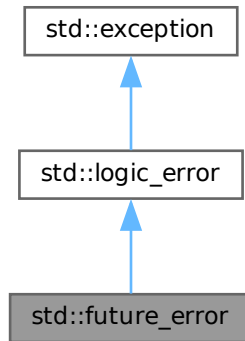
Retrieving the value.

The documentation for this class was generated from the following file:

- `future`

6.467 std::future_error Class Reference

Inheritance diagram for std::future_error:



Public Member Functions

- **future_error** ([future_errc](#) __errc)
- const [error_code](#) & **code** () const noexcept
- virtual const char * **what** () const noexcept

Friends

- void **__throw_future_error** (int)

6.467.1 Detailed Description

Exception type thrown by futures.

Since

C++11

6.467.2 Member Function Documentation

what()

```
virtual const char * std::future_error::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::logic_error](#).

The documentation for this class was generated from the following file:

- [future](#)

6.468 std::gamma_distribution<_RealType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- [gamma_distribution](#) ()
- [gamma_distribution](#) (_RealType __alpha_val, _RealType __beta_val=_RealType(1))
- [gamma_distribution](#) (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void [generate](#) ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType [alpha](#) () const
- _RealType [beta](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) [operator](#)() (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) [operator](#)() (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & [operator<<](#) ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::gamma_distribution](#)< _RealType1 > &__x)
- bool [operator==](#) (const [gamma_distribution](#) &__d1, const [gamma_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::gamma_distribution](#)< _RealType1 > &__x)

6.468.1 Detailed Description

template<typename _RealType = double>
class std::gamma_distribution<_RealType>

A gamma continuous distribution for random numbers.
 The formula for the gamma probability density function is:

$$p(x|\alpha, \beta) = \frac{1}{\beta\Gamma(\alpha)} (x/\beta)^{\alpha-1} e^{-x/\beta}$$

6.468.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::gamma_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.468.3 Constructor & Destructor Documentation

gamma_distribution() [1/2]

```
template<typename _RealType = double>
std::gamma_distribution< _RealType >::gamma_distribution ( ) [inline]
```

Constructs a gamma distribution with parameters 1 and 1.

gamma_distribution() [2/2]

```
template<typename _RealType = double>
std::gamma_distribution< _RealType >::gamma_distribution (
    _RealType __alpha_val,
    _RealType __beta_val = _RealType(1) ) [inline], [explicit]
```

Constructs a gamma distribution with parameters α and β .

6.468.4 Member Function Documentation

alpha()

```
template<typename _RealType = double>
_RealType std::gamma_distribution< _RealType >::alpha ( ) const [inline]
```

Returns the α of the distribution.

beta()

```
template<typename _RealType = double>
_RealType std::gamma_distribution< _RealType >::beta ( ) const [inline]
```

Returns the β of the distribution.

max()

```
template<typename _RealType = double>
result_type std::gamma_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits<_Tp>::max\(\)](#).

min()

```
template<typename _RealType = double>
result_type std::gamma_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>() [1/2]

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::gamma_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::gamma_distribution<_RealType>::operator\(\)\(\)](#).

Referenced by [std::gamma_distribution<_RealType>::operator\(\)\(\)](#).

operator()() [2/2]

```
template<typename _RealType >
template<typename _UniformRandomNumberGenerator >
gamma_distribution<_RealType >::result_type std::gamma_distribution<_RealType >::operator() (
    _UniformRandomNumberGenerator & __urng,
    const param_type & __param )
```

Marsaglia, G. and Tsang, W. W. "A Simple Method for Generating Gamma Variables" ACM Transactions on Mathematical Software, 26, 3, 363-372, 2000.

References [std::log\(\)](#), and [std::pow\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::gamma_distribution<_RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

Referenced by [std::chi_squared_distribution<_RealType>::param\(\)](#).

param() [2/2]

```
template<typename _RealType = double>
void std::gamma_distribution<_RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::gamma_distribution<_RealType >::reset ( ) [inline]
```

Resets the distribution state.

References [std::normal_distribution<_RealType>::reset\(\)](#).

Referenced by [std::chi_squared_distribution<_RealType>::reset\(\)](#), [std::fisher_f_distribution<_RealType>::reset\(\)](#), [std::student_t_distribution<_RealType>::reset\(\)](#), and [std::negative_binomial_distribution<_IntType>::reset\(\)](#).

6.468.5 Friends And Related Symbol Documentation

operator<<

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream<_CharT, _Traits > & operator<< (
    std::basic_ostream<_CharT, _Traits > & __os,
    const std::gamma_distribution<_RealType1 > & __x ) [friend]
```

Inserts a gamma_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>gamma_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const gamma\_distribution< _RealType > & __d1,
    const gamma\_distribution< _RealType > & __d2 ) [friend]
```

Return true if two gamma distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic\_istream< _CharT, _Traits > & operator>> (
    std::basic\_istream< _CharT, _Traits > & __is,
    std::gamma\_distribution< _RealType1 > & __x ) [friend]
```

Extracts a `gamma_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>gamma_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.469 `std::geometric_distribution<_IntType>` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **geometric_distribution** (const [param_type](#) &__p)
- **geometric_distribution** (double __p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- double **p** () const
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- bool **operator==** (const [geometric_distribution](#) &__d1, const [geometric_distribution](#) &__d2)

6.469.1 Detailed Description

```
template<typename _IntType = int>
class std::geometric_distribution<_IntType>
```

A discrete geometric random number distribution.

The formula for the geometric probability density function is $p(i|p) = p(1 - p)^i$ where p is the parameter of the distribution.

6.469.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::geometric\_distribution<\_IntType>::result\_type
The type of the range of the distribution.
```

6.469.3 Member Function Documentation

max()

```
template<typename _IntType = int>
result\_type std::geometric\_distribution<\_IntType>::max ( ) const [inline]
Returns the least upper bound value of the distribution.
References std::numeric\_limits<\_Tp>::max\(\).
```

min()

```
template<typename _IntType = int>
result_type std::geometric_distribution< _IntType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>()()

```
template<typename _IntType = int>
template<typename _UniformRandomNumberGenerator >
result_type std::geometric_distribution< _IntType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::geometric_distribution< _IntType >::operator>\(\)\(\)](#).

Referenced by [std::geometric_distribution< _IntType >::operator>\(\)\(\)](#).

p()

```
template<typename _IntType = int>
double std::geometric_distribution< _IntType >::p ( ) const [inline]
```

Returns the distribution parameter p.

param() [1/2]

```
template<typename _IntType = int>
param_type std::geometric_distribution< _IntType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

Referenced by [std::operator>>\(\)](#).

param() [2/2]

```
template<typename _IntType = int>
void std::geometric_distribution< _IntType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _IntType = int>
void std::geometric_distribution< _IntType >::reset ( ) [inline]
```

Resets the distribution state.

Does nothing for the geometric distribution.

6.469.4 Friends And Related Symbol Documentation**operator==**

```
template<typename _IntType = int>
bool operator== (
    const geometric_distribution< _IntType > & __d1,
    const geometric_distribution< _IntType > & __d2 ) [friend]
```

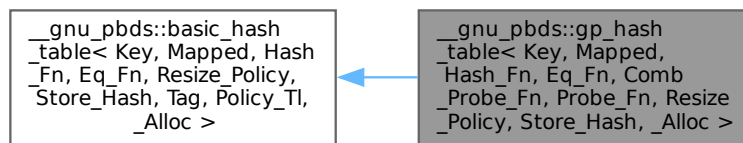
Return true if two geometric distributions have the same parameters.
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.470 `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >` Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >`:



Public Types

- typedef Comb_Probe_Fn **comb_probe_fn**
- typedef [gp_hash_tag](#) **container_category**
- typedef Eq_Fn **eq_fn**
- typedef Hash_Fn **hash_fn**
- typedef Probe_Fn **probe_fn**
- typedef Resize_Policy **resize_policy**

Public Member Functions

- [gp_hash_table](#) ()
- [gp_hash_table](#) (const [gp_hash_table](#) &other)
- [gp_hash_table](#) (const hash_fn &h)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p, const resize_policy &rp)
- template<typename It >
[gp_hash_table](#) (It first, It last)
- template<typename It >
[gp_hash_table](#) (It first, It last, const hash_fn &h)
- template<typename It >
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e)
- template<typename It >
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp)
- template<typename It >
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p)

- `template<typename It >`
`gp_hash_table` (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p, const resize_policy &rp)
- `gp_hash_table` & **operator=** (const `gp_hash_table` &other)
- void **swap** (`gp_hash_table` &other)

6.470.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_↵
_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool
Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
class __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_↵
Policy, Store_Hash, _Alloc >
```

A general-probing hash-based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Comb_Probe_Fn</i>	Combining probe functor. If Hash_Fn is not null_type, then this is the ranged-probe functor; otherwise, this is the range-hashing functor. XXX See Design::Hash-Based Containers::Hash Policies.
<i>Probe_Fn</i>	Probe functor.
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key. If Hash_Fn is null_type, then the container will not compile if this value is true
<i>_Alloc</i>	Allocator type.

Base tag choices are: `gp_hash_tag`.

Base is `basic_hash_table`.

6.470.2 Constructor & Destructor Documentation

`gp_hash_table()` [1/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_↵
Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>↵
::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table ( ) [inline]
```

Default constructor.

`gp_hash_table()` [2/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
```

```
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (
    const hash_fn & h ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object.

`gp_hash_table()` [3/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (
    const hash_fn & h,
    const eq_fn & e ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

`gp_hash_table()` [4/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object.

`gp_hash_table()` [5/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp,
    const probe_fn & p ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object, and `r_probe_fn` will be copied by the `probe_fn` object of the container object.

gp_hash_table() [6/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_↵
Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>↵
::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp,
    const probe_fn & p,
    const resize_policy & rp ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object, `r_probe_fn` will be copied by the `probe_fn` object of the container object, and `r_resize_policy` will be copied by the `Resize_Policy` object of the container object.

gp_hash_table() [7/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_↵
Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>↵
::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

gp_hash_table() [8/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_↵
Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>↵
::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last,
    const hash_fn & h ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object.

gp_hash_table() [9/12]

```
template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
```

```

detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash,
typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e ) [inline]

```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

`gp_hash_table()` [10/12]

```

template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash,
typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp ) [inline]

```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object.

`gp_hash_table()` [11/12]

```

template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash,
typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp,
    const probe_fn & p ) [inline]

```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object, and `r_probe_fn` will be copied by the `probe_fn` object of the container object.

gp_hash_table() [12/12]

```

template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<↵
Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_↵
Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>↵
::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table (
    It first,
    It last,
    const hash_fn & h,
    const eq_fn & e,
    const comb_probe_fn & cp,
    const probe_fn & p,
    const resize_policy & rp ) [inline]

```

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, r_comb_probe_fn will be copied by the comb_↵probe_fn object of the container object, r_probe_fn will be copied by the probe_fn object of the container object, and r_resize_policy will be copied by the resize_policy object of the container object.

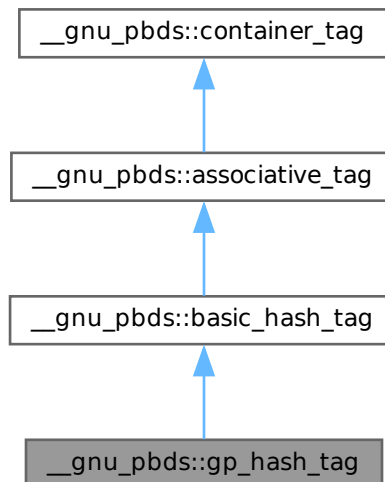
The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.471 __gnu_pbds::gp_hash_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::gp_hash_tag:



6.471.1 Detailed Description

General-probing hash.

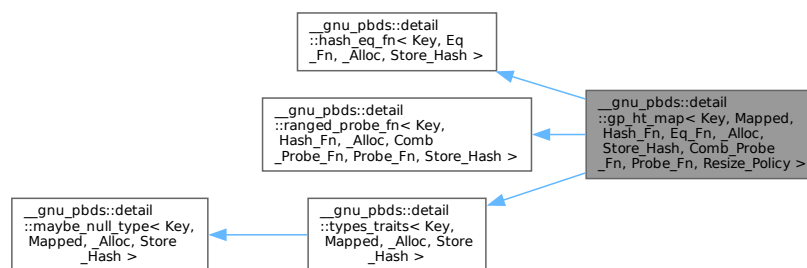
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.472 `__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >` Class Template Reference

```
#include <gp_ht_map_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >`:



Public Types

- enum { **store_hash** }
- typedef `_Alloc` **allocator_type**
- typedef `Comb_Probe_Fn` **comb_probe_fn**
- typedef `const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `Eq_Fn` **eq_fn**
- typedef `Hash_Fn` **hash_fn**
- typedef `iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**
- typedef `traits_base::key_const_reference` **key_const_reference**
- typedef `traits_base::key_pointer` **key_pointer**
- typedef `traits_base::key_reference` **key_reference**
- typedef `traits_base::key_type` **key_type**
- typedef `traits_base::mapped_const_pointer` **mapped_const_pointer**
- typedef `traits_base::mapped_const_reference` **mapped_const_reference**
- typedef `traits_base::mapped_pointer` **mapped_pointer**
- typedef `traits_base::mapped_reference` **mapped_reference**
- typedef `traits_base::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `point_const_iterator` **point_const_iterator**
- typedef `point_iterator` **point_iterator**

- typedef traits_base::pointer **pointer**
- typedef Probe_Fn **probe_fn**
- typedef traits_base::reference **reference**
- typedef Resize_Policy **resize_policy**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef stored_data< value_type, size_type, Store_Hash > **stored_data_type**
- typedef traits_base::value_type **value_type**

Public Member Functions

- **gp_ht_map** (const gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy > &)
- **gp_ht_map** (const Hash_Fn &)
- **gp_ht_map** (const Hash_Fn &, const Eq_Fn &)
- **gp_ht_map** (const Hash_Fn &, const Eq_Fn &, const Comb_Probe_Fn &)
- **gp_ht_map** (const Hash_Fn &, const Eq_Fn &, const Comb_Probe_Fn &, const Probe_Fn &)
- **gp_ht_map** (const Hash_Fn &, const Eq_Fn &, const Comb_Probe_Fn &, const Probe_Fn &, const Resize_Policy &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- point_iterator **find_end** ()
- point_const_iterator **find_end** () const
- Comb_Probe_Fn & **get_comb_probe_fn** ()
- const Comb_Probe_Fn & **get_comb_probe_fn** () const
- Eq_Fn & **get_eq_fn** ()
- const Eq_Fn & **get_eq_fn** () const
- Hash_Fn & **get_hash_fn** ()
- const Hash_Fn & **get_hash_fn** () const
- Probe_Fn & **get_probe_fn** ()
- const Probe_Fn & **get_probe_fn** () const
- Resize_Policy & **get_resize_policy** ()
- const Resize_Policy & **get_resize_policy** () const
- std::pair< point_iterator, bool > **insert** (const_reference r_val)
- size_type **max_size** () const
- mapped_reference **operator[]** (key_const_reference r_key)
- size_type **size** () const
- void **swap** (gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy > &)

Public Attributes

- no_throw_indicator `m_no_throw_copies_indicator`
- store_extra `m_store_extra_indicator`

Friends

- class `const_iterator_`
- class `iterator_`

6.472.1 Detailed Description

`template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool Store_Hash, typename Comb_Probe_Fn, typename Probe_Fn, typename Resize_Policy>`
`class __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >`

A general-probing hash-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor. Default is <code>__gnu_cxx::hash</code> .
<i>Eq_Fn</i>	Equal functor. Default <code>std::equal_to<Key></code>
<i>_Alloc</i>	Allocator type.
<i>Store_Hash</i>	If key type stores extra metadata. Defaults to false.
<i>Comb_Probe_Fn</i>	Combining probe functor. If <i>Hash_Fn</i> is not null_type, then this is the ranged-probe functor; otherwise, this is the range-hashing functor. XXX See Design::Hash-Based Containers::Hash Policies. Default <code>direct_mask_range_hashing</code> .
<i>Probe_Fn</i>	Probe functor. Defaults to <code>linear_probe_fn</code> , also <code>quadratic_probe_fn</code> .
<i>Resize_Policy</i>	Resizes hash. Defaults to <code>hash_standard_resize_policy</code> , using <code>hash_exponential_size_policy</code> and <code>hash_load_check_resize_trigger</code> .

Bases are: `detail::hash_eq_fn`, `Resize_Policy`, `detail::ranged_probe_fn`, `detail::types_traits`. (Optional: `detail::debug_map_base`.)

6.472.2 Member Enumeration Documentation

anonymous enum

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
anonymous enum
```

Value stores hash, true or false.

6.472.3 Member Function Documentation

`empty()`

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
bool __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_
Fn, Probe_Fn, Resize_Policy >::empty ( ) const [inline]
```


True if size() == 0.

get_comb_probe_fn() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
Comb_Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_comb_probe_fn ( )
Return current comb_probe_fn.
```

get_comb_probe_fn() [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
const Comb_Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_↵
Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_comb_probe_fn ( ) const
Return current const comb_probe_fn.
```

get_eq_fn() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
Eq_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_↵
Probe_Fn, Probe_Fn, Resize_Policy >::get_eq_fn ( )
Return current eq_fn.
```

get_eq_fn() [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
const Eq_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_eq_fn ( ) const
Return current const eq_fn.
```

get_hash_fn() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
Hash_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_↵
Probe_Fn, Probe_Fn, Resize_Policy >::get_hash_fn ( )
Return current hash_fn.
```

get_hash_fn() [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
const Hash_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_hash_fn ( ) const
Return current const hash_fn.
```

get_probe_fn() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
```

```
Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_probe_fn ( )
Return current probe_fn.
```

get_probe_fn() [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
const Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_probe_fn ( ) const
Return current const probe_fn.
```

get_resize_policy() [1/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
Resize_Policy & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_resize_policy ( )
Return current resize_policy.
```

get_resize_policy() [2/2]

```
template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc ,
bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy >
const Resize_Policy & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,
Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_resize_policy ( ) const
Return current const resize_policy.
```

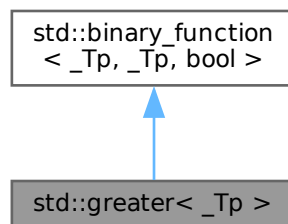
The documentation for this class was generated from the following file:

- [gp_ht_map.hpp](#)

6.473 std::greater<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::greater<_Tp>:

**Public Types**

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)

- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr bool **operator()** (const `_Tp` &__x, const `_Tp` &__y) const

6.473.1 Detailed Description

```
template<typename _Tp>
struct std::greater<_Tp>
```

One of the [comparison functors](#).

6.473.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary_function<_Tp, _Tp, bool>::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type [inherited]
second_argument_type is the type of the second argument
The documentation for this struct was generated from the following file:
```

- [stl_function.h](#)

6.474 std::ranges::greater Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using `is_transparent` = `__is_transparent`

Public Member Functions

- template<typename `_Tp`, typename `_Up`>
requires `totally_ordered_with<_Tp, _Up>`
constexpr bool **operator()** (`_Tp` &&__t, `_Up` &&__u) const noexcept(noexcept([std::declval](#)<`_Up`>())<
[std::declval](#)<`_Tp`>()))

6.474.1 Detailed Description

`ranges::greater` function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.475 std::greater< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward< _Tp >(__t) > std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t) > std::forward< _Up >(__u))
- template<typename _Tp, typename _Up >
constexpr bool **operator()** (_Tp *__t, _Up *__u) const noexcept

6.475.1 Detailed Description

One of the [comparison functions](#).

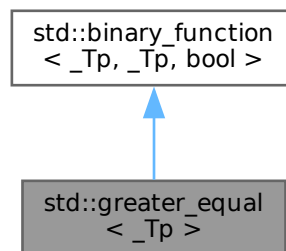
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.476 std::greater_equal< _Tp > Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::greater_equal< _Tp >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- constexpr bool **operator()** (const _Tp &__x, const _Tp &__y) const

6.476.1 Detailed Description

```
template<typename _Tp>
struct std::greater_equal< _Tp >
```

One of the [comparison functions](#).

6.476.2 Member Typedef Documentation

first_argument_type

`typedef _Tp std::binary_function< _Tp , _Tp , bool >::first_argument_type` [inherited]
`first_argument_type` is the type of the first argument

result_type

`typedef bool std::binary_function< _Tp , _Tp , bool >::result_type` [inherited]
`result_type` is the return type

second_argument_type

`typedef _Tp std::binary_function< _Tp , _Tp , bool >::second_argument_type` [inherited]
`second_argument_type` is the type of the second argument
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.477 `std::ranges::greater_equal` Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using `is_transparent` = `__is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >`
requires `totally_ordered_with<_Tp, _Up>`
`constexpr bool operator() (_Tp &&__t, _Up &&__u) const` `noexcept(noexcept(std::declval< _Tp >())<
std::declval< _Up >()))`

6.477.1 Detailed Description

`ranges::greater_equal` function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.478 `std::greater_equal< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >`
`constexpr auto operator() (_Tp &&__t, _Up &&__u) const` `noexcept(noexcept(std::forward< _Tp >(__t)↵
t) >=std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t) >=std::forward< _Up >(__u))`
- `template<typename _Tp, typename _Up >`
`constexpr bool operator() (_Tp *__t, _Up *__u) const` `noexcept`

6.478.1 Detailed Description

One of the [comparison functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.479 `__gnu_cxx::random_condition::group_adjustor` Struct Reference

```
#include <throw_allocator.h>
```

Inherits `__gnu_cxx::random_condition::adjustor_base`.

Public Member Functions

- `group_adjustor` (size_t size)

6.479.1 Detailed Description

Group condition.

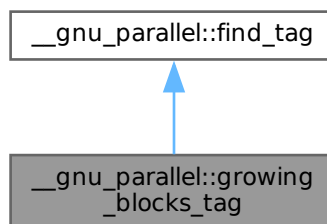
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.480 `__gnu_parallel::growing_blocks_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::growing_blocks_tag`:



6.480.1 Detailed Description

Selects the growing block size variant for `std::find()`.

See also

`_GLIBCXX_FIND_GROWING_BLOCKS`

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.481 `std::gslice` Class Reference

```
#include <gslice.h>
```

Public Member Functions

- [gslice](#) ()
- [gslice](#) (const [gslice](#) &)
- [gslice](#) (size_t __o, const [valarray](#)< size_t > &__l, const [valarray](#)< size_t > &__s)
- [~gslice](#) ()
- [gslice](#) & [operator=](#) (const [gslice](#) &)
- [valarray](#)< size_t > [size](#) () const
- size_t [start](#) () const
- [valarray](#)< size_t > [stride](#) () const

Friends

- template<typename _Tp >
class [valarray](#)

6.481.1 Detailed Description

Class defining multi-dimensional subset of an array.

The slice class represents a multi-dimensional subset of an array, specified by three parameter sets: start offset, size array, and stride array. The start offset is the index of the first element of the array that is part of the subset. The size and stride array describe each dimension of the slice. Size is the number of elements in that dimension, and stride is the distance in the array between successive elements in that dimension. Each dimension's size and stride is taken to begin at an array element described by the previous dimension. The size array and stride array must be the same size. For example, if you have offset==3, stride[0]==11, size[1]==3, stride[1]==3, then slice[0,0]==array[3], slice[0,1]==array[6], slice[0,2]==array[9], slice[1,0]==array[14], slice[1,1]==array[17], slice[1,2]==array[20].

The documentation for this class was generated from the following file:

- [gslice.h](#)

6.482 std::gslice_array< _Tp > Class Template Reference

```
#include <gslice_array.h>
```

Public Types

- typedef _Tp [value_type](#)

Public Member Functions

- [gslice_array](#) (const [gslice_array](#) &)
- template<class _Dom >
void [operator%=](#) (const _Expr< _Dom, _Tp > &) const
- void [operator%=](#) (const [valarray](#)< _Tp > &) const
- template<class _Dom >
void [operator&=](#) (const _Expr< _Dom, _Tp > &) const
- void [operator&=](#) (const [valarray](#)< _Tp > &) const
- template<class _Dom >
void [operator*%=](#) (const _Expr< _Dom, _Tp > &) const
- void [operator*%=](#) (const [valarray](#)< _Tp > &) const
- template<class _Dom >
void [operator*+=](#) (const _Expr< _Dom, _Tp > &) const
- void [operator*+=](#) (const [valarray](#)< _Tp > &) const
- template<class _Dom >
void [operator-=](#) (const _Expr< _Dom, _Tp > &) const

- void **operator=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator/=** (const _Expr<_Dom, _Tp> &) const
- void **operator/=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator<<=** (const _Expr<_Dom, _Tp> &) const
- void **operator<<=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator=** (const _Expr<_Dom, _Tp> &) const
- void **operator=** (const _Tp &) const
- [gslice_array](#) & **operator=** (const [gslice_array](#) &)
- void **operator=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator>>=** (const _Expr<_Dom, _Tp> &) const
- void **operator>>=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator^=** (const _Expr<_Dom, _Tp> &) const
- void **operator^=** (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void **operator|=** (const _Expr<_Dom, _Tp> &) const
- void **operator|=** (const [valarray](#)<_Tp> &) const

Friends

- class [valarray](#)<_Tp>

6.482.1 Detailed Description

template<typename _Tp>
class std::gslice_array<_Tp>

Reference to multi-dimensional subset of an array.

A [gslice_array](#) is a reference to the actual elements of an array specified by a [gslice](#). The way to get a [gslice_array](#) is to call `operator[](gslice)` on a [valarray](#). The returned [gslice_array](#) then permits carrying operations out on the referenced subset of elements in the original [valarray](#). For example, `operator+=(valarray)` will add values to the subset of elements in the underlying [valarray](#) this [gslice_array](#) refers to.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

6.482.2 Member Function Documentation

operator%=()

```
template<typename _Tp >
void std::gslice_array<_Tp>::operator%=(
    const valarray<_Tp> & ) const
```

Modulo slice elements by corresponding elements of *v*.

operator&=()

```
template<typename _Tp >
```



```
void std::gslice_array< _Tp >::operator&= (
    const valarray< _Tp > & ) const
```

Logical and slice elements with corresponding elements of *v*.

operator*=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator*= (
    const valarray< _Tp > & ) const
```

Multiply slice elements by corresponding elements of *v*.

operator+=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator+= (
    const valarray< _Tp > & ) const
```

Add corresponding elements of *v* to slice elements.

operator-=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator-= (
    const valarray< _Tp > & ) const
```

Subtract corresponding elements of *v* from slice elements.

operator/=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator/= (
    const valarray< _Tp > & ) const
```

Divide slice elements by corresponding elements of *v*.

operator<<=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator<<= (
    const valarray< _Tp > & ) const
```

Left shift slice elements by corresponding elements of *v*.

operator>>=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator>>= (
    const valarray< _Tp > & ) const
```

Right shift slice elements by corresponding elements of *v*.

operator^=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator^= (
    const valarray< _Tp > & ) const
```

Logical xor slice elements with corresponding elements of *v*.

operator" |=()

```
template<typename _Tp >
void std::gslice_array< _Tp >::operator|= (
    const valarray< _Tp > & ) const
```

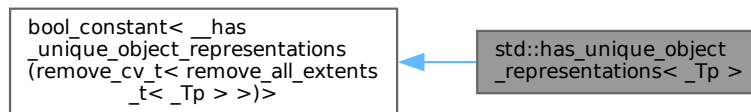
Logical or slice elements with corresponding elements of v.

The documentation for this class was generated from the following files:

- [valarray](#)
- [gslice_array.h](#)

6.483 std::has_unique_object_representations< _Tp > Struct Template Reference

Inheritance diagram for std::has_unique_object_representations< _Tp >:

**Public Types**

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.483.1 Detailed Description

```
template<typename _Tp>
struct std::has_unique_object_representations< _Tp >
```

has_unique_object_representations

Since

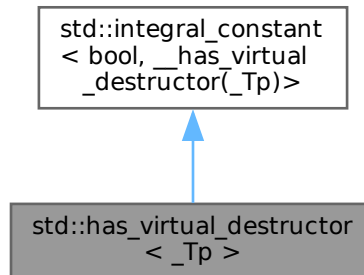
C++17

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.484 `std::has_virtual_destructor<_Tp>` Struct Template Reference

Inheritance diagram for `std::has_virtual_destructor<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.484.1 Detailed Description

```
template<typename _Tp>
struct std::has_virtual_destructor<_Tp>
```

has_virtual_destructor

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.485 `std::hash<_Tp>` Struct Template Reference

```
#include <functional_hash.h>
```

Inherits `std::__hash_enum<_Tp, bool>`.

6.485.1 Detailed Description

```
template<typename _Tp>
struct std::hash<_Tp>
```

Primary class template hash.

Primary class template hash, usable for enum types only.
 The documentation for this struct was generated from the following file:

- [string_view](#)

6.486 `std::hash< __debug::bitset< _Nb > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__debug::bitset< _Nb >](#) &__b) const noexcept

6.486.1 Detailed Description

```
template<size_t _Nb>
struct std::hash< __debug::bitset< _Nb > >
```

`std::hash` specialization for `bitset`.
 The documentation for this struct was generated from the following file:

- [debug/bitset](#)

6.487 `std::hash< __debug::vector< bool, _Alloc > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__debug::vector< bool, _Alloc >](#) &__b) const noexcept

6.487.1 Detailed Description

```
template<typename _Alloc>
struct std::hash< __debug::vector< bool, _Alloc > >
```

`std::hash` specialization for `vector<bool>`.
 The documentation for this struct was generated from the following file:

- [debug/vector](#)

6.488 `std::hash< __gnu_cxx::__u16vstring >` Struct Reference

```
#include <vstring.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__gnu_cxx::__u16vstring](#) &__s) const noexcept

6.488.1 Detailed Description

std::hash specialization for `__u16vstring`.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

6.489 std::hash< __gnu_cxx::__u32vstring > Struct Reference

```
#include <vstring.h>
```

Inherits std::__hash_base< `_Result`, `_Arg` >.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__gnu_cxx::__u32vstring](#) &__s) const noexcept

6.489.1 Detailed Description

std::hash specialization for `__u32vstring`.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

6.490 std::hash< __gnu_cxx::__vstring > Struct Reference

```
#include <vstring.h>
```

Inherits std::__hash_base< `_Result`, `_Arg` >.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__gnu_cxx::__vstring](#) &__s) const noexcept

6.490.1 Detailed Description

std::hash specialization for `__vstring`.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

6.491 std::hash< __gnu_cxx::__wvstring > Struct Reference

```
#include <vstring.h>
```

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (const __gnu_cxx::__wvstring &__s) const noexcept

6.491.1 Detailed Description

std::hash specialization for __wvstring.

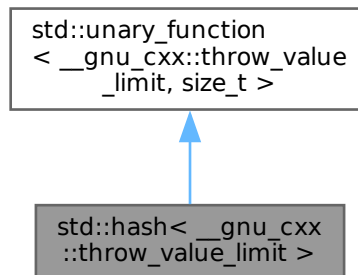
The documentation for this struct was generated from the following file:

- [vstring.h](#)

6.492 std::hash< __gnu_cxx::throw_value_limit > Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for std::hash< __gnu_cxx::throw_value_limit >:



Public Types

- typedef __gnu_cxx::throw_value_limit **argument_type**
- typedef size_t **result_type**

Public Member Functions

- size_t **operator()** (const __gnu_cxx::throw_value_limit &__val) const

6.492.1 Detailed Description

Explicit specialization of std::hash for __gnu_cxx::throw_value_limit.

6.492.2 Member Typedef Documentation

argument_type

```
typedef __gnu_cxx::throw_value_limit std::unary_function< __gnu_cxx::throw_value_limit , size_t
>::argument_type [inherited]
```

argument_type is the type of the argument

result_type

```
typedef size_t std::unary_function< __gnu_cxx::throw_value_limit , size_t >::result_type [inherited]
```

result_type is the return type

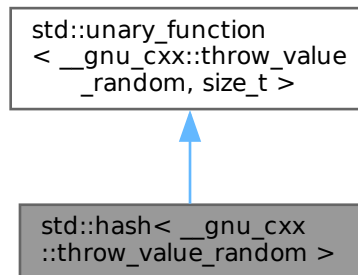
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.493 std::hash< __gnu_cxx::throw_value_random > Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for std::hash< __gnu_cxx::throw_value_random >:



Public Types

- typedef `__gnu_cxx::throw_value_random` `argument_type`
- typedef `size_t` `result_type`

Public Member Functions

- `size_t operator()` (const `__gnu_cxx::throw_value_random` &__val) const

6.493.1 Detailed Description

Explicit specialization of `std::hash` for `__gnu_cxx::throw_value_random`.

6.493.2 Member Typedef Documentation

argument_type

```
typedef __gnu_cxx::throw_value_random std::unary_function< __gnu_cxx::throw_value_random , size_t
>::argument_type [inherited]
```

`argument_type` is the type of the argument

result_type

```
typedef size_t std::unary_function< __gnu_cxx::throw_value_random , size_t >::result_type [inherited]
```

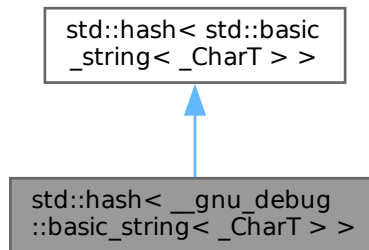
`result_type` is the return type

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.494 `std::hash< __gnu_debug::basic_string< _CharT > >` Struct Template Reference

Inheritance diagram for `std::hash< __gnu_debug::basic_string< _CharT > >`:



6.494.1 Detailed Description

```
template<typename _CharT>
struct std::hash< __gnu_debug::basic_string< _CharT > >
```

`std::hash` specialization for `__gnu_debug::basic_string`.

The documentation for this struct was generated from the following file:

- [debug/string](#)

6.495 `std::hash< __shared_ptr< _Tp, _Lp > >` Struct Template Reference

```
#include <shared_ptr_base.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const `__shared_ptr< _Tp, _Lp > &__s`) const noexcept

6.495.1 Detailed Description

```
template<typename _Tp, _Lock_policy _Lp>
struct std::hash< __shared_ptr< _Tp, _Lp > >
```

std::hash specialization for __shared_ptr.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

6.496 std::hash< _Tp * > Struct Template Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- **size_t operator()** (_Tp *__p) const noexcept

6.496.1 Detailed Description

```
template<typename _Tp>
struct std::hash< _Tp * >
```

Partial specializations for pointer types.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.497 std::hash< bool > Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- **size_t operator()** (bool __val) const noexcept

6.497.1 Detailed Description

Explicit specialization for bool.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.498 `std::hash< char >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- `typedef _Arg argument_type`
- `typedef _Result result_type`

Public Member Functions

- `size_t operator() (char __val) const` noexcept

6.498.1 Detailed Description

Explicit specialization for `char`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.499 `std::hash< char16_t >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- `typedef _Arg argument_type`
- `typedef _Result result_type`

Public Member Functions

- `size_t operator() (char16_t __val) const` noexcept

6.499.1 Detailed Description

Explicit specialization for `char16_t`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.500 `std::hash< char32_t >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- `typedef _Arg argument_type`
- `typedef _Result result_type`

Public Member Functions

- `size_t operator() (char32_t __val) const` noexcept

6.500.1 Detailed Description

Explicit specialization for `char32_t`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.501 `std::hash< double >` Struct Reference

```
#include <functional_hash.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`double __val`) `const noexcept`

6.501.1 Detailed Description

Specialization for `double`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.502 `std::hash< error_code >` Struct Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const error_code &__e`) `const noexcept`

6.502.1 Detailed Description

`std::hash` specialization for `error_code`.

The documentation for this struct was generated from the following file:

- [system_error](#)

6.503 `std::hash< error_condition >` Struct Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [error_condition](#) &__e) const noexcept

6.503.1 Detailed Description

`std::hash` specialization for `error_condition`.

The documentation for this struct was generated from the following file:

- [system_error](#)

6.504 `std::hash< experimental::optional< _Tp > >` Struct Template Reference**Public Types**

- using `argument_type` = [experimental::optional< _Tp >](#)
- using `result_type` = `size_t`

Public Member Functions

- `size_t operator()` (const [experimental::optional< _Tp >](#) &__t) const noexcept(noexcept([hash< _Tp >](#) {}(*__t)))

6.504.1 Detailed Description

`template<typename _Tp>`

`struct std::hash< experimental::optional< _Tp > >`

`std::hash` partial specialization for `experimental::optional`

The documentation for this struct was generated from the following file:

- [experimental/optional](#)

6.505 `std::hash< experimental::shared_ptr< _Tp > >` Struct Template Reference

```
#include <shared_ptr.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` `argument_type`
- typedef `_Result` `result_type`

Public Member Functions

- `size_t operator()` (const `experimental::shared_ptr< _Tp >` &__s) const noexcept

6.505.1 Detailed Description

`template<typename _Tp>`

`struct std::hash< experimental::shared_ptr< _Tp > >`

`std::hash` specialization for `shared_ptr`.

The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

6.506 `std::hash< float >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`float __val`) `const noexcept`

6.506.1 Detailed Description

Specialization for `float`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.507 `std::hash< int >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`int __val`) `const noexcept`

6.507.1 Detailed Description

Explicit specialization for `int`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.508 `std::hash< long >` Struct Reference

```
#include <functional_hash.h>  
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`long __val`) `const noexcept`

6.508.1 Detailed Description

Explicit specialization for long.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.509 `std::hash< long double >` Struct Reference

```
#include <functional_hash.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`long double __val`) `const noexcept`

6.509.1 Detailed Description

Specialization for long double.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.510 `std::hash< long long >` Struct Reference

```
#include <functional_hash.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`long long __val`) `const noexcept`

6.510.1 Detailed Description

Explicit specialization for long long.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.511 `std::hash< shared_ptr< _Tp > >` Struct Template Reference

```
#include <shared_ptr.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [shared_ptr](#)< `_Tp` > &__s) const noexcept

6.511.1 Detailed Description

```
template<typename _Tp>
struct std::hash< shared_ptr< _Tp > >
```

std::hash specialization for shared_ptr.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

6.512 std::hash< short > Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (short __val) const noexcept

6.512.1 Detailed Description

Explicit specialization for short.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.513 std::hash< signed char > Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (signed char __val) const noexcept

6.513.1 Detailed Description

Explicit specialization for signed char.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.514 `std::hash< string >` Struct Reference

```
#include <basic_string.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [string](#) &__s) const noexcept

6.514.1 Detailed Description

`std::hash` specialization for string.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

6.515 `std::hash< thread::id >` Struct Reference

```
#include <std_thread.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [thread::id](#) &__id) const noexcept

6.515.1 Detailed Description

`std::hash` specialization for `thread::id`.

The documentation for this struct was generated from the following file:

- [std_thread.h](#)

6.516 `std::hash< type_index >` Struct Reference

Public Types

- typedef [type_index](#) **argument_type**
- typedef `size_t` **result_type**

Public Member Functions

- `size_t operator()` (const [type_index](#) &__ti) const noexcept

6.516.1 Detailed Description

std::hash specialization for type_index.

The documentation for this struct was generated from the following file:

- [typeindex](#)

6.517 std::hash< u16string > Struct Reference

```
#include <basic_string.h>
```

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- `size_t operator()` (const [u16string](#) &__s) const noexcept

6.517.1 Detailed Description

std::hash specialization for u16string.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

6.518 std::hash< u32string > Struct Reference

```
#include <basic_string.h>
```

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- `size_t operator()` (const [u32string](#) &__s) const noexcept

6.518.1 Detailed Description

std::hash specialization for u32string.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

6.519 std::hash< unique_ptr< _Tp, _Dp > > Struct Template Reference

```
#include <unique_ptr.h>
```

Inherits std::__hash_base< _Result, _Arg >, and __uniq_ptr_hash< unique_ptr< _Tp, _Dp > >.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

6.519.1 Detailed Description

```
template<typename _Tp, typename _Dp>
struct std::hash< unique_ptr< _Tp, _Dp > >
```

`std::hash` specialization for `unique_ptr`.

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

6.520 `std::hash< unsigned char >` Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`unsigned char __val`) `const noexcept`

6.520.1 Detailed Description

Explicit specialization for `unsigned char`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.521 `std::hash< unsigned int >` Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`unsigned int __val`) `const noexcept`

6.521.1 Detailed Description

Explicit specialization for `unsigned int`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.522 `std::hash< unsigned long >` Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`unsigned long __val`) `const noexcept`

6.522.1 Detailed Description

Explicit specialization for unsigned long.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.523 `std::hash< unsigned long long >` Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`unsigned long long __val`) `const noexcept`

6.523.1 Detailed Description

Explicit specialization for unsigned long long.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.524 `std::hash< unsigned short >` Struct Reference

```
#include <functional_hash.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`unsigned short __val`) `const noexcept`

6.524.1 Detailed Description

Explicit specialization for unsigned short.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.525 `std::hash< wchar_t >` Struct Reference

```
#include <functional_hash.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`wchar_t __val`) const noexcept

6.525.1 Detailed Description

Explicit specialization for `wchar_t`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

6.526 `std::hash< wstring >` Struct Reference

```
#include <basic_string.h>
```

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [wstring](#) &__s) const noexcept

6.526.1 Detailed Description

`std::hash` specialization for `wstring`.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

6.527 `std::hash<::bitset< _Nb > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const ::bitset< _Nb > &__b`) `const noexcept`

6.527.1 Detailed Description

```
template<size_t _Nb>
struct std::hash<::bitset< _Nb > >
```

`std::hash` specialization for `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

6.528 `std::hash<::vector< bool, _Alloc > >` Struct Template Reference

```
#include <stl_bvector.h>
Inherits std::__hash_base< _Result, _Arg >.
```

Public Types

- `typedef _Arg argument_type`
- `typedef _Result result_type`

Public Member Functions

- `size_t operator()` (`const ::vector< bool, _Alloc > &`) `const noexcept`

6.528.1 Detailed Description

```
template<typename _Alloc>
struct std::hash<::vector< bool, _Alloc > >
```

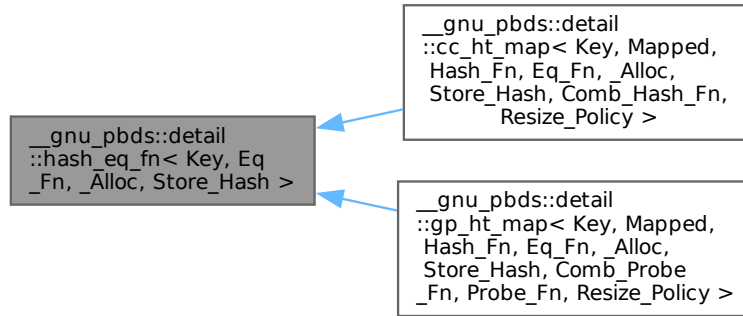
`std::hash` specialization for `vector<bool>`.

The documentation for this struct was generated from the following files:

- [stl_bvector.h](#)
- [vector.tcc](#)

6.529 `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >`:



6.529.1 Detailed Description

```
template<typename Key, typename Eq_Fn, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >
```

Primary template.

The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

6.530 `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false >` Struct Template Reference

```
#include <hash_eq_fn.hpp>
Inherits Eq_Fn.
```

Public Types

- typedef `Eq_Fn` **eq_fn_base**
- typedef [rebind_traits](#)< `_Alloc`, `Key` >::const_reference **key_const_reference**

Public Member Functions

- **hash_eq_fn** (`const Eq_Fn &r_eq_fn`)
- **bool operator()** (`key_const_reference r_lhs_key`, `key_const_reference r_rhs_key`) `const`
- **void swap** (`const hash_eq_fn &other`)

6.530.1 Detailed Description

```
template<typename Key, typename Eq_Fn, typename _Alloc>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false >
```

Specialization 1 - The client requests that hash values not be stored.

The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

6.531 `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true >` Struct Template Reference

`#include <hash_eq_fn.hpp>`

Inherits `Eq_Fn`.

Public Types

- typedef `Eq_Fn` **eq_fn_base**
- typedef [rebind_traits](#)< `_Alloc`, `Key` >::const_reference **key_const_reference**
- typedef `_Alloc::size_type` **size_type**

Public Member Functions

- **hash_eq_fn** (`const Eq_Fn &r_eq_fn`)
- bool **operator()** (`key_const_reference r_lhs_key`, `size_type lhs_hash`, `key_const_reference r_rhs_key`, `size_type rhs_hash`) const
- void **swap** (`const hash_eq_fn &other`)

6.531.1 Detailed Description

```
template<typename Key, class Eq_Fn, class _Alloc>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true >
```

Specialization 2 - The client requests that hash values be stored.

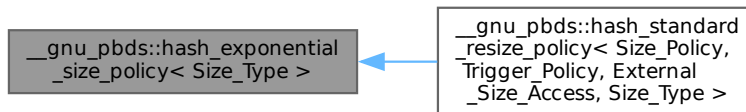
The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

6.532 `__gnu_pbds::hash_exponential_size_policy< Size_Type >` Class Template Reference

`#include <hash_policy.hpp>`

Inheritance diagram for `__gnu_pbds::hash_exponential_size_policy< Size_Type >`:



Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- [hash_exponential_size_policy](#) (`size_type start_size=8`, `size_type grow_factor=2`)
- void **swap** (`hash_exponential_size_policy< Size_Type > &other`)

Protected Member Functions

- `size_type` `get_nearest_larger_size` (`size_type` `size`) `const`
- `size_type` `get_nearest_smaller_size` (`size_type` `size`) `const`

6.532.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::hash_exponential_size_policy< Size_Type >
```

A size policy whose sequence of sizes form an exponential sequence (typically powers of 2).

6.532.2 Constructor & Destructor Documentation

`hash_exponential_size_policy()`

```
template<typename Size_Type = std::size_t>
__gnu_pbds::hash_exponential_size_policy< Size_Type >::hash_exponential_size_policy (
    size_type start_size = 8,
    size_type grow_factor = 2 )
```

Default constructor, or onstructor taking a `start_size`, or constructor taking a `start_size` and `grow_factor`. The policy will use the sequence of sizes `start_size`, `start_size* grow_factor`, `start_size* grow_factor^2`, ...

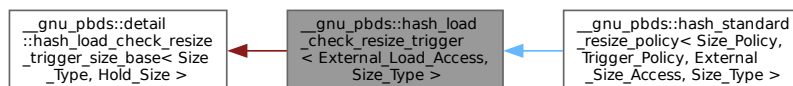
The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.533 `__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >` Class Template Reference

```
#include <hash_policy.hpp>
```

Inheritance diagram for `__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >`:



Public Types

- enum { `external_load_access` }
- typedef `Size_Type` `size_type`

Public Member Functions

- [hash_load_check_resize_trigger](#) (`float` `load_min`=0.125, `float` `load_max`=0.5)
- `std::pair< float, float >` `get_loads` () `const`
- void `set_loads` (`std::pair< float, float >` `load_pair`)
- void `swap` (`hash_load_check_resize_trigger` &`other`)

Protected Member Functions

- bool **is_grow_needed** (size_type size, size_type num_entries) const
- bool **is_resize_needed** () const
- void **notify_cleared** ()
- void **notify_erase_search_collision** ()
- void **notify_erase_search_end** ()
- void **notify_erase_search_start** ()
- void **notify_erased** (size_type num_entries)
- void **notify_externally_resized** (size_type new_size)
- void **notify_find_search_collision** ()
- void **notify_find_search_end** ()
- void **notify_find_search_start** ()
- void **notify_insert_search_collision** ()
- void **notify_insert_search_end** ()
- void **notify_insert_search_start** ()
- void **notify_inserted** (size_type num_entries)
- void **notify_resized** (size_type new_size)

6.533.1 Detailed Description

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >
```

A resize trigger policy based on a load check. It keeps the load factor between some load factors load_min and load_max.

6.533.2 Member Enumeration Documentation

anonymous enum

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
anonymous enum
```

Enumerator

external_load_access	Specifies whether the load factor can be accessed externally. The two options have different trade-offs in terms of flexibility, genericity, and encapsulation.
----------------------	---

6.533.3 Constructor & Destructor Documentation

hash_load_check_resize_trigger()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::hash_load_check_↵
resize_trigger (
    float load_min = 0.125,
    float load_max = 0.5 )
```

Default constructor, or constructor taking load_min and load_max load factors between which this policy will keep the actual load.

6.533.4 Member Function Documentation

get_loads()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
std::pair< float, float > __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::get_loads ( ) const [inline]
```

Returns a pair of the minimal and maximal loads, respectively.

notify_cleared()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::notify_cleared ( ) [protected]
```

Notifies the table was cleared.

notify_inserted()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::notify_inserted (
    size_type num_entries ) [inline], [protected]
```

Notifies an element was inserted. The total number of entries in the table is num_entries.

notify_resized()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::notify_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized as a result of this object's signifying that a resize is needed.

set_loads()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::set_loads (
    std::pair< float, float > load_pair )
```

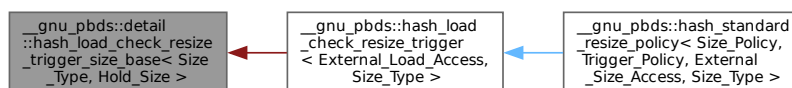
Sets the loads through a pair of the minimal and maximal loads, respectively.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.534 `__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >`:



6.534.1 Detailed Description

```
template<typename Size_Type, bool Hold_Size>
class __gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >
```

Primary template.

The documentation for this class was generated from the following file:

- [hash_load_check_resize_trigger_size_base.hpp](#)

6.535 __gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true > Class Template Reference

```
#include <hash_load_check_resize_trigger_size_base.hpp>
```

Protected Types

- typedef Size_Type **size_type**

Protected Member Functions

- size_type **get_size** () const
- void **set_size** (size_type size)
- void **swap** ([hash_load_check_resize_trigger_size_base](#) &other)

6.535.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >
```

Specializations.

The documentation for this class was generated from the following file:

- [hash_load_check_resize_trigger_size_base.hpp](#)

6.536 __gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc > Class Template Reference

Public Types

- typedef _Ht::allocator_type **allocator_type**
- typedef _Ht::const_iterator **const_iterator**
- typedef _Ht::const_pointer **const_pointer**
- typedef _Ht::const_reference **const_reference**
- typedef _Tp **data_type**
- typedef _Ht::difference_type **difference_type**
- typedef _Ht::hasher **hasher**
- typedef _Ht::iterator **iterator**
- typedef _Ht::key_equal **key_equal**
- typedef _Ht::key_type **key_type**
- typedef _Tp **mapped_type**
- typedef _Ht::pointer **pointer**
- typedef _Ht::reference **reference**
- typedef _Ht::size_type **size_type**
- typedef _Ht::value_type **value_type**

Public Member Functions

- `template<class _InputIterator>`
hash_map (`_InputIterator __f, _InputIterator __l`)
- `template<class _InputIterator>`
hash_map (`_InputIterator __f, _InputIterator __l, size_type __n`)
- `template<class _InputIterator>`
hash_map (`_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf`)
- `template<class _InputIterator>`
hash_map (`_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type()`)
- **hash_map** (`size_type __n`)
- **hash_map** (`size_type __n, const hasher &__hf`)
- **hash_map** (`size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type()`)
- iterator **begin** ()
- const_iterator **begin** () const
- size_type **bucket_count** () const
- void **clear** ()
- size_type **count** (`const key_type &__key`) const
- size_type **elems_in_bucket** (`size_type __n`) const
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- `pair< iterator, iterator >` **equal_range** (`const key_type &__key`)
- `pair< const_iterator, const_iterator >` **equal_range** (`const key_type &__key`) const
- size_type **erase** (`const key_type &__key`)
- void **erase** (`iterator __f, iterator __l`)
- void **erase** (`iterator __it`)
- iterator **find** (`const key_type &__key`)
- const_iterator **find** (`const key_type &__key`) const
- allocator_type **get_allocator** () const
- hasher **hash_funct** () const
- `template<class _InputIterator>`
void **insert** (`_InputIterator __f, _InputIterator __l`)
- `pair< iterator, bool >` **insert** (`const value_type &__obj`)
- `pair< iterator, bool >` **insert_noresize** (`const value_type &__obj`)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- `_Tp &` **operator[]** (`const key_type &__key`)
- void **resize** (`size_type __hint`)
- size_type **size** () const
- void **swap** (`hash_map &__hs`)

Friends

- `template<class _K1, class _T1, class _HF, class _EqK, class _AI>`
bool **operator==** (`const hash_map< _K1, _T1, _HF, _EqK, _AI > &, const hash_map< _K1, _T1, _HF, _EqK, _AI > &`)

6.536.1 Detailed Description

```
template<class _Key, class _Tp, class _HashFn = hash<_Key>, class _EqualKey = equal_to<_Key>, class
_Alloc = allocator<_Tp>>
class __gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

The documentation for this class was generated from the following file:

- [hash_map](#)

6.537 __gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc > Class Template Reference

Public Types

- typedef _Ht::allocator_type **allocator_type**
- typedef _Ht::const_iterator **const_iterator**
- typedef _Ht::const_pointer **const_pointer**
- typedef _Ht::const_reference **const_reference**
- typedef _Tp **data_type**
- typedef _Ht::difference_type **difference_type**
- typedef _Ht::hasher **hasher**
- typedef _Ht::iterator **iterator**
- typedef _Ht::key_equal **key_equal**
- typedef _Ht::key_type **key_type**
- typedef _Tp **mapped_type**
- typedef _Ht::pointer **pointer**
- typedef _Ht::reference **reference**
- typedef _Ht::size_type **size_type**
- typedef _Ht::value_type **value_type**

Public Member Functions

- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__↵_eq, const allocator_type &__a=allocator_type())
- **hash_multimap** (size_type __n)
- **hash_multimap** (size_type __n, const hasher &__hf)
- **hash_multimap** (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__↵a=allocator_type())
- iterator **begin** ()
- const_iterator **begin** () const
- size_type **bucket_count** () const

- void **clear** ()
- size_type **count** (const key_type &__key) const
- size_type **elems_in_bucket** (size_type __n) const
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- pair< iterator, iterator > **equal_range** (const key_type &__key)
- pair< const_iterator, const_iterator > **equal_range** (const key_type &__key) const
- size_type **erase** (const key_type &__key)
- void **erase** (iterator __f, iterator __l)
- void **erase** (iterator __it)
- iterator **find** (const key_type &__key)
- const_iterator **find** (const key_type &__key) const
- allocator_type **get_allocator** () const
- hasher **hash_func** () const
- template<class _InputIterator >
void **insert** (_InputIterator __f, _InputIterator __l)
- iterator **insert** (const value_type &__obj)
- iterator **insert_noresize** (const value_type &__obj)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- void **resize** (size_type __hint)
- size_type **size** () const
- void **swap** (hash_multimap &__hs)

Friends

- template<class _K1, class _T1, class _HF, class _EqK, class _AI >
bool **operator==** (const hash_multimap< _K1, _T1, _HF, _EqK, _AI > &, const hash_multimap< _K1, _T1, _HF, _EqK, _AI > &)

6.537.1 Detailed Description

```
template<class _Key, class _Tp, class _HashFn = hash<_Key>, class _EqualKey = equal_to<_Key>, class
_Alloc = allocator<_Tp>>
class __gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

The documentation for this class was generated from the following file:

- [hash_map](#)

6.538 `__gnu_cxx::hash_multiset<_Value, _HashFcn, _EqualKey, _Alloc>` Class Template Reference

Public Types

- typedef _Ht::allocator_type **allocator_type**
- typedef _Ht::const_iterator **const_iterator**

- typedef _Alloc::const_pointer **const_pointer**
- typedef _Alloc::const_reference **const_reference**
- typedef _Ht::difference_type **difference_type**
- typedef _Ht::hasher **hasher**
- typedef _Ht::const_iterator **iterator**
- typedef _Ht::key_equal **key_equal**
- typedef _Ht::key_type **key_type**
- typedef _Alloc::pointer **pointer**
- typedef _Alloc::reference **reference**
- typedef _Ht::size_type **size_type**
- typedef _Ht::value_type **value_type**

Public Member Functions

- template<class _InputIterator >
hash_multiset (_InputIterator __f, _InputIterator __l)
- template<class _InputIterator >
hash_multiset (_InputIterator __f, _InputIterator __l, size_type __n)
- template<class _InputIterator >
hash_multiset (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)
- template<class _InputIterator >
hash_multiset (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- **hash_multiset** (size_type __n)
- **hash_multiset** (size_type __n, const hasher &__hf)
- **hash_multiset** (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- iterator **begin** () const
- size_type **bucket_count** () const
- void **clear** ()
- size_type **count** (const key_type &__key) const
- size_type **elems_in_bucket** (size_type __n) const
- bool **empty** () const
- iterator **end** () const
- pair< iterator, iterator > **equal_range** (const key_type &__key) const
- size_type **erase** (const key_type &__key)
- void **erase** (iterator __f, iterator __l)
- void **erase** (iterator __it)
- iterator **find** (const key_type &__key) const
- allocator_type **get_allocator** () const
- hasher **hash_funct** () const
- template<class _InputIterator >
void **insert** (_InputIterator __f, _InputIterator __l)
- iterator **insert** (const value_type &__obj)
- iterator **insert_noresize** (const value_type &__obj)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- void **resize** (size_type __hint)
- size_type **size** () const
- void **swap** ([hash_multiset](#) &hs)

Friends

- `template<class _Val, class _HF, class _EqK, class _Al >`
`bool operator== (const hash_multiset< _Val, _HF, _EqK, _Al > &, const hash_multiset< _Val, _HF, _EqK, _Al > &)`

6.538.1 Detailed Description

`template<class _Value, class _HashFcn = hash<_Value>, class _EqualKey = equal_to<_Value>, class _Alloc = allocator<_Value>>`
`class __gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc >`

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

The documentation for this class was generated from the following file:

- [hash_set](#)

6.539 __gnu_pbds::hash_prime_size_policy Class Reference

```
#include <hash_policy.hpp>
```

Public Types

- `typedef std::size_t size_type`

Public Member Functions

- `hash_prime_size_policy (size_type start_size=8)`
- `void swap (hash_prime_size_policy &other)`

Protected Member Functions

- `size_type get_nearest_larger_size (size_type size) const`
- `size_type get_nearest_smaller_size (size_type size) const`

6.539.1 Detailed Description

A size policy whose sequence of sizes form a nearly-exponential sequence of primes.

6.539.2 Member Typedef Documentation

[size_type](#)

```
typedef std::size_t __gnu_pbds::hash_prime_size_policy::size_type
```

Size type.

6.539.3 Constructor & Destructor Documentation

[hash_prime_size_policy](#)()

```
__gnu_pbds::hash_prime_size_policy::hash_prime_size_policy (
    size\_type start_size = 8 )
```

Default constructor, or onstructor taking a start_size The policy will use the sequence of sizes approximately start_size, start_size* 2, start_size* 2^2, ...

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.540 `__gnu_cxx::hash_set<_Value, _HashFcn, _EqualKey, _Alloc >` Class Template Reference

Public Types

- typedef `_Ht::allocator_type` **allocator_type**
- typedef `_Ht::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `_Ht::difference_type` **difference_type**
- typedef `_Ht::hasher` **hasher**
- typedef `_Ht::const_iterator` **iterator**
- typedef `_Ht::key_equal` **key_equal**
- typedef `_Ht::key_type` **key_type**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `_Ht::size_type` **size_type**
- typedef `_Ht::value_type` **value_type**

Public Member Functions

- template<class `_InputIterator` >
hash_set (`_InputIterator __f, _InputIterator __l`)
- template<class `_InputIterator` >
hash_set (`_InputIterator __f, _InputIterator __l, size_type __n`)
- template<class `_InputIterator` >
hash_set (`_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf`)
- template<class `_InputIterator` >
hash_set (`_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type()`)
- **hash_set** (`size_type __n`)
- **hash_set** (`size_type __n, const hasher &__hf`)
- **hash_set** (`size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type()`)
- iterator **begin** () const
- `size_type` **bucket_count** () const
- void **clear** ()
- `size_type` **count** (const `key_type &__key`) const
- `size_type` **elems_in_bucket** (`size_type __n`) const
- bool **empty** () const
- iterator **end** () const
- [pair](#)< iterator, iterator > **equal_range** (const `key_type &__key`) const
- `size_type` **erase** (const `key_type &__key`)
- void **erase** (iterator `__f, iterator __l`)
- void **erase** (iterator `__it`)
- iterator **find** (const `key_type &__key`) const
- `allocator_type` **get_allocator** () const
- hasher **hash_funct** () const
- template<class `_InputIterator` >
void **insert** (`_InputIterator __f, _InputIterator __l`)

- `pair< iterator, bool > insert` (const value_type &__obj)
- `pair< iterator, bool > insert_noresize` (const value_type &__obj)
- `key_equal key_eq` () const
- `size_type max_bucket_count` () const
- `size_type max_size` () const
- `void resize` (size_type __hint)
- `size_type size` () const
- `void swap` (hash_set &__hs)

Friends

- `template<class _Val, class _HF, class _EqK, class _AI >`
`bool operator==` (const hash_set< _Val, _HF, _EqK, _AI > &, const hash_set< _Val, _HF, _EqK, _AI > &)

6.540.1 Detailed Description

`template<class _Value, class _HashFcn = hash<_Value>, class _EqualKey = equal_to<_Value>, class _Alloc = allocator<_Value>>`

`class __gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc >`

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

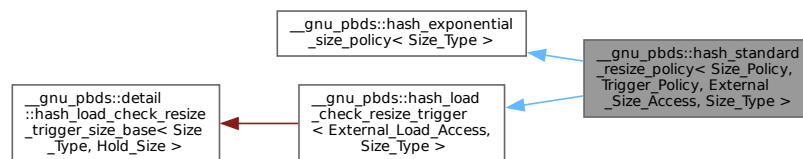
The documentation for this class was generated from the following file:

- `hash_set`

6.541 `__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >` Class Template Reference

`#include <hash_policy.hpp>`

Inheritance diagram for `__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >`:



Public Types

- enum { `external_load_access` }
- enum { `external_size_access` }
- typedef `Size_Policy` `size_policy`
- typedef `Size_Type` `size_type`
- typedef `Trigger_Policy` `trigger_policy`

Public Member Functions

- [hash_standard_resize_policy](#) ()
- [hash_standard_resize_policy](#) (const Size_Policy &r_size_policy)
- [hash_standard_resize_policy](#) (const Size_Policy &r_size_policy, const Trigger_Policy &r_trigger_policy)
- size_type [get_actual_size](#) () const
- [std::pair](#)< float, float > [get_loads](#) () const
- Size_Policy & [get_size_policy](#) ()
- const Size_Policy & [get_size_policy](#) () const
- Trigger_Policy & [get_trigger_policy](#) ()
- const Trigger_Policy & [get_trigger_policy](#) () const
- void [resize](#) (size_type suggested_new_size)
- void [set_loads](#) ([std::pair](#)< float, float > load_pair)
- void [swap](#) ([hash_exponential_size_policy](#)< Size_Type > &other)
- void [swap](#) ([hash_load_check_resize_trigger](#) &other)
- void [swap](#) ([hash_standard_resize_policy](#)< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type > &other)

Protected Member Functions

- size_type [get_nearest_larger_size](#) (size_type size) const
- size_type [get_nearest_smaller_size](#) (size_type size) const
- size_type [get_new_size](#) (size_type size, size_type num_used_e) const
- bool [is_grow_needed](#) (size_type size, size_type num_entries) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) (size_type num_e)
- void [notify_externally_resized](#) (size_type new_size)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) (size_type num_e)
- void [notify_resized](#) (size_type new_size)

6.541.1 Detailed Description

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_load_↵
check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_↵
Type >
```

A resize policy which delegates operations to size and trigger policies.

6.541.2 Member Enumeration Documentation

anonymous enum

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
anonymous enum [inherited]
```

Enumerator

<code>external_load_access</code>	Specifies whether the load factor can be accessed externally. The two options have different trade-offs in terms of flexibility, genericity, and encapsulation.
-----------------------------------	---

6.541.3 Constructor & Destructor Documentation

`hash_standard_resize_policy()` [1/3]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_↵
_Type >::hash_standard_resize_policy ( )
```

Default constructor.

`hash_standard_resize_policy()` [2/3]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_↵
_Type >::hash_standard_resize_policy (
    const Size_Policy & r_size_policy )
```

constructor taking some policies `r_size_policy` will be copied by the `Size_Policy` object of this object.

`hash_standard_resize_policy()` [3/3]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_↵
_Type >::hash_standard_resize_policy (
    const Size_Policy & r_size_policy,
    const Trigger_Policy & r_trigger_policy )
```

constructor taking some policies. `r_size_policy` will be copied by the `Size_Policy` object of this object. `r_trigger_policy` will be copied by the `Trigger_Policy` object of this object.

6.541.4 Member Function Documentation

`get_actual_size()`

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
size_type __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_↵
Access, Size_Type >::get_actual_size ( ) const [inline]
```

Returns the actual size of the container.

`get_loads()`

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
std::pair< float, float > __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_↵
_Type >::get_loads ( ) const [inline], [inherited]
```

Returns a pair of the minimal and maximal loads, respectively.

get_new_size()

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
size_type __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_↵
Access, Size_Type >::get_new_size (
    size_type size,
    size_type num_used_e ) const [protected]
```

Queries what the new size should be, when the container is resized naturally. The current `__size` of the container is `size`, and the number of used entries within the container is `num_used_e`.

get_size_policy() [1/2]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
Size_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_↵
Size_Access, Size_Type >::get_size_policy ( )
```

Access to the `Size_Policy` object used.

get_size_policy() [2/2]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
const Size_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_↵
_Size_Access, Size_Type >::get_size_policy ( ) const
```

Const access to the `Size_Policy` object used.

get_trigger_policy() [1/2]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
Trigger_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_↵
Size_Access, Size_Type >::get_trigger_policy ( )
```

Access to the `Trigger_Policy` object used.

get_trigger_policy() [2/2]

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
const Trigger_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy,
External_Size_Access, Size_Type >::get_trigger_policy ( ) const
```

Access to the `Trigger_Policy` object used.

resize()

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_↵
load_check_resize_trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access,
Size_Type >::resize (
    size_type suggested_new_size )
```

Resizes the container to `suggested_new_size`, a suggested size (the actual size will be determined by the `Size_Policy` object).

set_loads()

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::set_loads (
    std::pair< float, float > load_pair ) [inherited]
```

Sets the loads through a pair of the minimal and maximal loads, respectively.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.542 std::locale::id Class Reference

```
#include <locale_classes.h>
```

Public Member Functions

- [id\(\)](#)
- [size_t _M_id\(\)](#) const throw ()

Friends

- template<typename _Facet >
bool [has_facet](#) (const [locale](#) &) throw ()
- class **locale**
- class **locale::_Impl**
- template<typename _Facet >
const _Facet & [use_facet](#) (const [locale](#) &)

6.542.1 Detailed Description

Facet ID class.

The ID class provides facets with an index used to identify them. Every facet class must define a public static member `locale::id`, or be derived from a facet that provides this member, otherwise the facet cannot be used in a locale. The `locale::id` ensures that each class type gets a unique identifier.

6.542.2 Constructor & Destructor Documentation**id()**

```
std::locale::id::id ( ) [inline]
Constructor.
```

6.542.3 Friends And Related Symbol Documentation**has_facet**

```
template<typename _Facet >
bool has_facet (
    const locale & ) throw ( ) [friend]
```

Test for the presence of a facet.

`has_facet` tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

use_facet

```
template<typename _Facet >
const _Facet & use_facet (
    const locale & ) [friend]
```

Return a facet.

`use_facet` looks for and returns a reference to a facet of type `Facet` where `Facet` is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type `Facet`.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

Reference to facet of type `Facet`.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

The documentation for this class was generated from the following file:

- [locale_classes.h](#)

6.543 std::thread::id Class Reference

```
#include <thread>
```

Public Member Functions

- `id (native_handle_type __id)`

Friends

- struct `hash< id >`
- bool `operator< (id __x, id __y)` noexcept
- template<class `_CharT`, class `_Traits` >
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, id __id)`

- bool **operator==** ([id __x](#), [id __y](#)) noexcept
- class **thread**

Related Symbols

(Note that these are not member symbols.)

- bool **operator==** ([thread::id __x](#), [thread::id __y](#)) noexcept
- bool **operator!=** ([thread::id __x](#), [thread::id __y](#)) noexcept
- bool **operator<** ([thread::id __x](#), [thread::id __y](#)) noexcept
- bool **operator<=** ([thread::id __x](#), [thread::id __y](#)) noexcept
- bool **operator>** ([thread::id __x](#), [thread::id __y](#)) noexcept
- bool **operator>=** ([thread::id __x](#), [thread::id __y](#)) noexcept
- template<class _CharT, class _Traits >
[basic_ostream](#)< _CharT, _Traits > & **operator<<** ([basic_ostream](#)< _CharT, _Traits > &__out, [thread::id __id](#))

6.543.1 Detailed Description

A std::thread::id is a unique identifier for a thread.

Since

C++11

The documentation for this class was generated from the following files:

- [std_thread.h](#)
- [thread](#)

6.544 std::identity Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using **is_transparent** = __is_transparent

Public Member Functions

- template<typename _Tp >
constexpr _Tp && **operator()** (_Tp &&__t) const noexcept

6.544.1 Detailed Description

[func.identity] The identity function.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.545 std::experimental::fundamentals_v1::in_place_t Struct Reference

6.545.1 Detailed Description

Tag type for in-place construction.

The documentation for this struct was generated from the following file:

- [experimental/optional](#)

6.546 `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >` Class Template Reference

```
#include <random.h>
```

Public Types

- typedef `_UIntType` `result_type`

Public Member Functions

- `independent_bits_engine` ()
- `independent_bits_engine` (`_RandomNumberEngine` && `__rng`)
- `template<typename _Sseq, typename = _If_seed_seq<_Sseq>>`
`independent_bits_engine` (`_Sseq` & `__q`)
- `independent_bits_engine` (`const _RandomNumberEngine` & `__rng`)
- `independent_bits_engine` (`result_type` `__s`)
- `const _RandomNumberEngine` & `base` () `const noexcept`
- `void discard` (`unsigned long long` `__z`)
- `result_type operator()` ()
- `void seed` ()
- `template<typename _Sseq >`
`_If_seed_seq<_Sseq>` `seed` (`_Sseq` & `__q`)
- `void seed` (`result_type` `__s`)

Static Public Member Functions

- `static constexpr result_type max` ()
- `static constexpr result_type min` ()

Friends

- `bool operator==` (`const independent_bits_engine` & `__lhs`, `const independent_bits_engine` & `__rhs`)
- `template<typename _CharT, typename _Traits >`
`std::basic_istream<_CharT, _Traits >` & `operator>>` (`std::basic_istream<_CharT, _Traits >` & `__is`,
`std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >` & `__x`)

6.546.1 Detailed Description

```
template<typename _RandomNumberEngine, size_t __w, typename _UIntType>
class std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >
```

Produces random numbers by combining random numbers from some base engine to produce random numbers with a specified number of bits `__w`.

6.546.2 Member Typedef Documentation

`result_type`

```
template<typename _RandomNumberEngine, size_t __w, typename _UIntType >
typedef _UIntType std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >::result_type
```

The type of the generated random value.

6.546.3 Constructor & Destructor Documentation

independent_bits_engine() [1/5]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::independent_bits_engine ( )
[inline]
```

Constructs a default independent_bits_engine engine.
The underlying engine is default constructed as well.

independent_bits_engine() [2/5]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::independent_bits_engine (
    const _RandomNumberEngine & __rng ) [inline], [explicit]
```

Copy constructs a independent_bits_engine engine.
Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

independent_bits_engine() [3/5]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::independent_bits_engine (
    _RandomNumberEngine && __rng ) [inline], [explicit]
```

Move constructs a independent_bits_engine engine.
Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

independent_bits_engine() [4/5]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::independent_bits_engine (
    result_type __s ) [inline], [explicit]
```

Seed constructs a independent_bits_engine engine.
Constructs the underlying generator engine seeded with __s.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

independent_bits_engine() [5/5]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
```

```
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::independent_bits_engine (
    _Sseq & __q ) [inline], [explicit]
```

Generator construct a independent_bits_engine engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

6.546.4 Member Function Documentation

base()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
const _RandomNumberEngine & std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >↵
::base ( ) const [inline], [noexcept]
```

Gets a const reference to the underlying generator engine object.

discard()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
void std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::discard (
    unsigned long long __z ) [inline]
```

Discard a sequence of random numbers.

max()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
static constexpr result_type std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType
>::max ( ) [inline], [static], [constexpr]
```

Gets the maximum value in the generated random number range.

min()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
static constexpr result_type std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType
>::min ( ) [inline], [static], [constexpr]
```

Gets the minimum value in the generated random number range.

operator>()

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::result_type std::independent_bits_engine<
_RandomNumberEngine, __w, _UIntType >::operator()
```

Gets the next value in the generated random number sequence.

References `std::__lg()`, `std::numeric_limits< _Tp >::max()`, and `std::numeric_limits< _Tp >::min()`.

seed() [1/3]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
void std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::seed ( ) [inline]
```

Reseeds the independent_bits_engine object with the default seed for the underlying base class generator engine.

seed() [2/3]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
template<typename _Sseq >
_If_seed_seq< _Sseq > std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::seed
(
    _Sseq & __q ) [inline]
```

Reseeds the independent_bits_engine object with the given seed sequence.

Parameters

<code>__q</code>	A seed generator function.
------------------	----------------------------

seed() [3/3]

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
void std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::seed (
    result_type __s ) [inline]
```

Reseeds the independent_bits_engine object with the default seed for the underlying base class generator engine.

6.546.5 Friends And Related Symbol Documentation**operator==**

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
bool operator== (
    const independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __lhs,
    const independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __rhs ) [friend]
```

Compares two independent_bits_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A independent_bits_engine random number generator object.
<code>__rhs</code>	Another independent_bits_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

operator>>

```
template<typename _RandomNumberEngine , size_t __w, typename _UIntType >
template<typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
-------------------	------------------

Parameters

<code>_↔</code>	A independent_bits_engine random number generator engine.
<code>_X</code>	

Returns

The input stream with the state of `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.547 std::indirect_array<_Tp> Class Template Reference

```
#include <indirect_array.h>
```

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- [indirect_array](#) (const [indirect_array](#) &)
- template<class `_Dom` >
void **operator%=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator%=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator&=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator&=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator*=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator*=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator+=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator+=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator-=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator-=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator/=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator/=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator<<=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator<<=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator=](#) (const `_Tp` &) const
- [indirect_array](#) & [operator=](#) (const [indirect_array](#) &)
- void [operator=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator>>=** (const `_Expr`< `_Dom`, `_Tp` > &) const

- void **operator>>=** (const **valarray**<_Tp> &) const
- template<class _Dom>
void **operator^=** (const _Expr<_Dom, _Tp> &) const
- void **operator^=** (const **valarray**<_Tp> &) const
- template<class _Dom>
void **operator|=** (const _Expr<_Dom, _Tp> &) const
- void **operator|=** (const **valarray**<_Tp> &) const

Friends

- class **gslice_array**<_Tp>
- class **valarray**<_Tp>

6.547.1 Detailed Description

template<class _Tp>
class std::indirect_array<_Tp>

Reference to arbitrary subset of an array.

An **indirect_array** is a reference to the actual elements of an array specified by an ordered array of indices. The way to get an **indirect_array** is to call **operator[]**(**valarray**<size_t>) on a **valarray**. The returned **indirect_array** then permits carrying operations out on the referenced subset of elements in the original **valarray**.

For example, if an **indirect_array** is obtained using the array (4,2,0) as an argument, and then assigned to an array containing (1,2,3), then the underlying array will have **array[0]==3**, **array[2]==2**, and **array[4]==1**.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

6.547.2 Member Function Documentation

operator%=()

```
template<class _Tp>
void std::indirect_array<_Tp>::operator%= (
    const valarray<_Tp> & ) const
```

Modulo slice elements by corresponding elements of *v*.

operator&=()

```
template<class _Tp>
void std::indirect_array<_Tp>::operator&= (
    const valarray<_Tp> & ) const
```

Logical and slice elements with corresponding elements of *v*.

operator*=()

```
template<class _Tp>
void std::indirect_array<_Tp>::operator*= (
    const valarray<_Tp> & ) const
```

Multiply slice elements by corresponding elements of *v*.

operator+=()

```
template<class _Tp>
```

```
void std::indirect_array< _Tp >::operator+= (
    const valarray< _Tp > & ) const
```

Add corresponding elements of *v* to slice elements.

operator-=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator-= (
    const valarray< _Tp > & ) const
```

Subtract corresponding elements of *v* from slice elements.

operator/=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator/= (
    const valarray< _Tp > & ) const
```

Divide slice elements by corresponding elements of *v*.

operator<<=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator<<= (
    const valarray< _Tp > & ) const
```

Left shift slice elements by corresponding elements of *v*.

operator>>=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator>>= (
    const valarray< _Tp > & ) const
```

Right shift slice elements by corresponding elements of *v*.

operator^=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator^= (
    const valarray< _Tp > & ) const
```

Logical xor slice elements with corresponding elements of *v*.

operator" |=()

```
template<class _Tp >
void std::indirect_array< _Tp >::operator|= (
    const valarray< _Tp > & ) const
```

Logical or slice elements with corresponding elements of *v*.

The documentation for this class was generated from the following files:

- [valarray](#)
- [indirect_array.h](#)

6.548 std::initializer_list< _E > Class Template Reference

Public Types

- typedef const _E * **const_iterator**
- typedef const _E & **const_reference**

- typedef const _E * **iterator**
- typedef const _E & **reference**
- typedef size_t **size_type**
- typedef _E **value_type**

Public Member Functions

- constexpr const_iterator **begin** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr size_type **size** () const noexcept

Related Symbols

(Note that these are not member symbols.)

- template<class _Tp >
constexpr const _Tp * **begin** (initializer_list< _Tp > __ils) noexcept
- template<class _Tp >
constexpr const _Tp * **end** (initializer_list< _Tp > __ils) noexcept

6.548.1 Detailed Description

template<class _E>

class std::initializer_list<_E>

initializer_list


6.548.2 Friends And Related Symbol Documentation

begin()

```
template<class _Tp >
constexpr const _Tp * begin (
    initializer_list< _Tp > __ils ) [related]
```

Return an iterator pointing to the first element of the initializer_list.

Parameters


 <code>__ils</code>	Initializer list.
--	-------------------

end()

```
template<class _Tp >
constexpr const _Tp * end (
    initializer_list< _Tp > __ils ) [related]
```

Return an iterator pointing to one past the last element of the initializer_list.

Parameters

 <code>__ils</code>	Initializer list.
--	-------------------

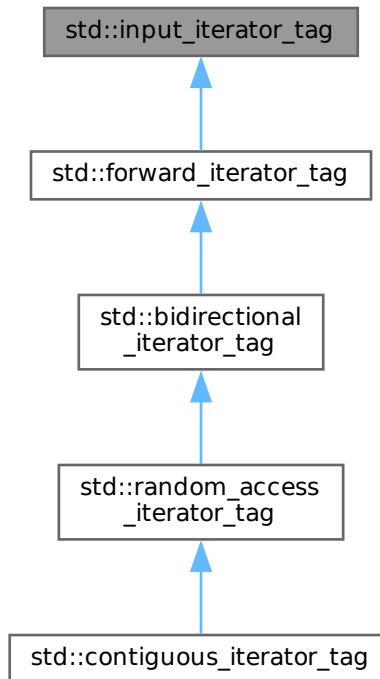
The documentation for this class was generated from the following file:

- [initializer_list](#)

6.549 std::input_iterator_tag Struct Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for std::input_iterator_tag:



6.549.1 Detailed Description

Marking input iterators.

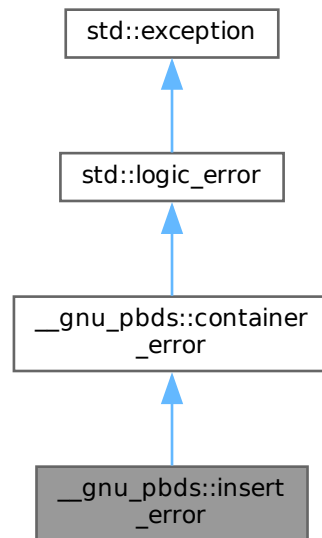
The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.550 __gnu_pbds::insert_error Struct Reference

```
#include <exception.hpp>
```

Inheritance diagram for `__gnu_pbds::insert_error`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.550.1 Detailed Description

An entry cannot be inserted into a container object for logical reasons (not, e.g., if memory is unavailable, in which case the `allocator_type`'s exception will be thrown).

6.550.2 Member Function Documentation

`what()`

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

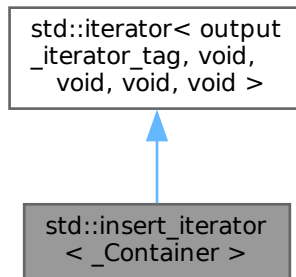
The documentation for this struct was generated from the following file:

- [exception.hpp](#)

6.551 `std::insert_iterator<_Container>` Class Template Reference

```
#include <stl_iterator.h>
```

Inheritance diagram for `std::insert_iterator< _Container >`:



Public Types

- typedef `_Container` `container_type`
- using `difference_type` = `ptrdiff_t`
- typedef `output_iterator_tag` `iterator_category`
- typedef void `pointer`
- typedef void `reference`
- typedef void `value_type`

Public Member Functions

- constexpr `insert_iterator` (`_Container` &__x, `_Iter` __i)
- constexpr `insert_iterator` & `operator*` ()
- constexpr `insert_iterator` & `operator++` ()
- constexpr `insert_iterator` & `operator++` (int)
- constexpr `insert_iterator` & `operator=` (const typename `_Container::value_type` &__value)
- constexpr `insert_iterator` & `operator=` (typename `_Container::value_type` &&__value)

Protected Attributes

- `_Container` * `container`
- `_Iter` `iter`

6.551.1 Detailed Description

```
template<typename _Container>
class std::insert_iterator< _Container >
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator inserts it in the container at the iterator's position, rather than overwriting the value at that position.

(Sequences will actually insert a *copy* of the value before the iterator's position.)

Tip: Using the `inserter` function to create these iterators can save typing.

6.551.2 Member Typedef Documentation

container_type

```
template<typename _Container >
typedef _Container std::insert_iterator< _Container >::container_type
```

A nested typedef for the type of whatever container you used.

iterator_category

```
typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >←
::iterator_category [inherited]
```

One of the [tag types](#).

pointer

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer [inherited]
```

This type represents a pointer-to-value_type.

reference

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference [inherited]
```

This type represents a reference-to-value_type.

value_type

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type [inherited]
```

The type "pointed to" by the iterator.

6.551.3 Constructor & Destructor Documentation

insert_iterator()

```
template<typename _Container >
constexpr std::insert_iterator< _Container >::insert_iterator (
    _Container & __x,
    _Iter __i ) [inline], [constexpr]
```

The only way to create this iterator is with a container and an initial position (a normal iterator into the container).

6.551.4 Member Function Documentation

operator*()

```
template<typename _Container >
constexpr insert_iterator & std::insert_iterator< _Container >::operator* ( ) [inline], [constexpr]
```

Simply returns *this.

operator++() [1/2]

```
template<typename _Container >
constexpr insert_iterator & std::insert_iterator< _Container >::operator++ ( ) [inline], [constexpr]
```

Simply returns *this. (This iterator does not *move*.)

operator++() [2/2]

```
template<typename _Container >
constexpr insert\_iterator & std::insert\_iterator< _Container >::operator++ (
    int ) [inline], [constexpr]
```

Simply returns *this. (This iterator does not *move*.)

operator=()

```
template<typename _Container >
constexpr insert\_iterator & std::insert\_iterator< _Container >::operator= (
    const typename _Container::value_type & __value ) [inline], [constexpr]
```

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const T for <code>container<T></code> .
----------------------	---

Returns

This iterator, for chained operations.

This kind of iterator maintains its own position in the container. Assigning a value to the iterator will insert the value into the container at the place before the iterator.

The position is maintained such that subsequent assignments will insert values immediately after one another. For example,

```
// vector v contains A and Z
insert\_iterator i (v, ++v.begin());
i = 1;
i = 2;
i = 3;

// vector v contains A, 1, 2, 3, and Z
```

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.552 std::integer_sequence<_Tp, _Idx> Struct Template Reference

```
#include <utility.h>
```

Public Types

- typedef `_Tp value_type`

Static Public Member Functions

- static constexpr `size_t size` () noexcept

6.552.1 Detailed Description

```
template<typename _Tp, _Tp... _Idx>
struct std::integer\_sequence< _Tp, _Idx >
```

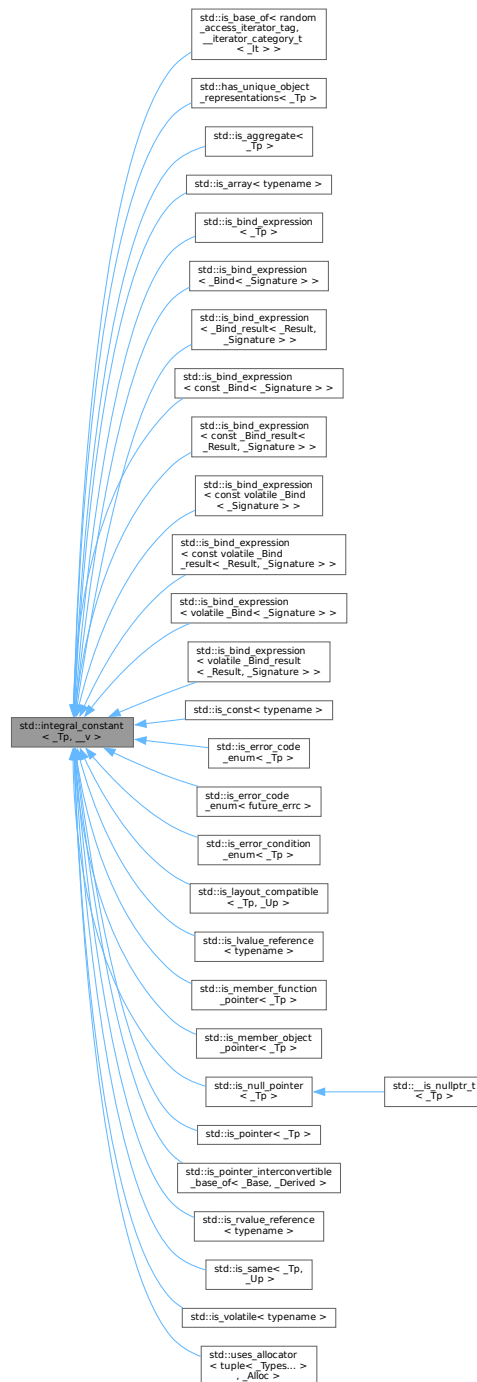
Class template `integer_sequence`.

The documentation for this struct was generated from the following file:

- [utility.h](#)

6.553 std::integral_constant< _Tp, __v > Struct Template Reference

Inheritance diagram for std::integral_constant< _Tp, __v >:



Public Types

- typedef `integral_constant< _Tp, __v >` type

- typedef `_Tp value_type`

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `_Tp value`

6.553.1 Detailed Description

```
template<typename _Tp, _Tp __v>
struct std::integral_constant<_Tp, __v >
```

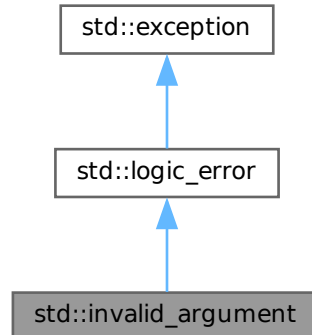
integral_constant

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.554 std::invalid_argument Class Reference

Inheritance diagram for std::invalid_argument:



Public Member Functions

- **invalid_argument** (const char *) _GLIBCXX_TXN_SAFE
- **invalid_argument** (const [invalid_argument](#) &)=default
- **invalid_argument** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **invalid_argument** ([invalid_argument](#) &&)=default
- [invalid_argument](#) & **operator=** (const [invalid_argument](#) &)=default
- [invalid_argument](#) & **operator=** ([invalid_argument](#) &&)=default
- virtual const char * **what** () const noexcept

Classes

- class [failure](#)

Public Types

- enum [event](#) { [erase_event](#) , [imbue_event](#) , [copyfmt_event](#) }
- typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
- typedef _ios_Fmtflags [fmtflags](#)
- typedef _ios_istate [iostate](#)
- typedef _ios_Openmode [openmode](#)
- typedef _ios_Seekdir [seekdir](#)

Public Member Functions

- [ios_base](#) (const [ios_base](#) &)=delete
- virtual [~ios_base](#) ()
- const [locale](#) & [_M_getloc](#) () const
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [locale](#) [getloc](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc) throw ()
- long & [iword](#) (int __ix)
- [ios_base](#) & [operator=](#) (const [ios_base](#) &)=delete
- [streamsize](#) [precision](#) () const
- [streamsize](#) [precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)
- void [register_callback](#) ([event_callback](#) __fn, int __index)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
- void [unsetf](#) ([fmtflags](#) __mask)
- [streamsize](#) [width](#) () const
- [streamsize](#) [width](#) ([streamsize](#) __wide)

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [openmode](#) [__noreplace](#)
- static const [fmtflags](#) [adjustfield](#)
- static const [openmode](#) [app](#)
- static const [openmode](#) [ate](#)
- static const [iostate](#) [badbit](#)
- static const [fmtflags](#) [basefield](#)
- static const [seekdir](#) [beg](#)
- static const [openmode](#) [binary](#)
- static const [fmtflags](#) [boolalpha](#)
- static const [seekdir](#) [cur](#)
- static const [fmtflags](#) [dec](#)
- static const [seekdir](#) [end](#)

- static const [iostate eofbit](#)
- static const [iostate failbit](#)
- static const [fmtflags fixed](#)
- static const [fmtflags floatfield](#)
- static const [iostate goodbit](#)
- static const [fmtflags hex](#)
- static const [openmode in](#)
- static const [fmtflags internal](#)
- static const [fmtflags left](#)
- static const [fmtflags oct](#)
- static const [openmode out](#)
- static const [fmtflags right](#)
- static const [fmtflags scientific](#)
- static const [fmtflags showbase](#)
- static const [fmtflags showpoint](#)
- static const [fmtflags showpos](#)
- static const [fmtflags skipws](#)
- static const [openmode trunc](#)
- static const [fmtflags unitbuf](#)
- static const [fmtflags uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_call_callbacks](#) ([event __ev](#)) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- [_Words & _M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- void [_M_move](#) ([ios_base &](#)) noexcept
- void [_M_swap](#) ([ios_base &__rhs](#)) noexcept

Protected Attributes

- [_Callback_list * _M_callbacks](#)
- [iostate _M_exception](#)
- [fmtflags _M_flags](#)
- [locale _M_ios_locale](#)
- [_Words _M_local_word](#) [[_S_local_word_size](#)]
- [streamsize _M_precision](#)
- [iostate _M_streambuf_state](#)
- [streamsize _M_width](#)
- [_Words * _M_word](#)
- int [_M_word_size](#)
- [_Words _M_word_zero](#)

6.556.1 Detailed Description

The base of the I/O class hierarchy.

This class defines everything that can be defined about I/O that does not depend on the type of characters being input or output. Most people will only see `ios_base` when they need to specify the full name of the various I/O flags (e.g., the openmodes).




6.556.2 Member Typedef Documentation

event_callback

```
typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i)
```

The type of an event callback function.

Parameters

 <code>__e</code>	One of the members of the event enum.
 <code>__b</code>	Reference to the ios_base object.
 <code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

fmtflags

```
typedef _Ios_Fmtflags std::ios_base::fmtflags
```

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type fmtflags are:

- boolalpha
- dec
- fixed
- hex
- internal
- left
- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

iostate

```
typedef _Ios_Iostate std::ios_base::iostate
```

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

openmode

```
typedef _Ios_Openmode std::ios_base::openmode
```

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

seekdir

```
typedef _Ios_Seekdir std::ios_base::seekdir
```

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

6.556.3 Member Enumeration Documentation

event

```
enum std::ios_base::event
```

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

6.556.4 Constructor & Destructor Documentation

`~ios_base()`

```
virtual std::ios_base::~~ios_base ( ) [virtual]
```

Invokes each callback with `erase_event`. Destroys local storage.

Note that the `ios_base` object for the standard streams never gets destroyed. As a result, any callbacks registered with the standard streams will not get invoked with `erase_event` (unless `copyfmt` is used).

6.556.5 Member Function Documentation

`_M_getloc()`

```
const locale & std::ios_base::_M_getloc ( ) const [inline]
```

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Referenced by `std::money_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_date()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::time_put<_CharT, _OutIter>::do_put()`, `std::time_get<_CharT, _InIter>::get()`, and `std::time_put<_CharT, _OutIter>::put()`.

`flags()` [1/2]

```
fmtflags std::ios_base::flags ( ) const [inline]
```

Access to format flags.

Returns

The format control flags for both input and output.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::operator<<()`, `std::operator>>()`, `std::operator>>()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, `std::operator>>()`, and `std::operator>>()`.

`flags()` [2/2]

```
fmtflags std::ios_base::flags (
    fmtflags __fmtfl ) [inline]
```

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

`getloc()`

```
locale std::ios_base::getloc ( ) const [inline]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#), [std::money_put<_CharT, _Outiter>::do_put\(\)](#), [std::operator<>\(\)](#), [std::operator<>\(\)](#), and [std::ws\(\)](#).

imbue()

```
locale std::ios_base::imbue (
    const locale & __loc ) throw ( )
```

Setting a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Sets the new locale for this stream, and then invokes each callback with `imbue_event`.

Referenced by [std::basic_ios<_CharT, _Traits>::imbue\(\)](#).

iword()

```
long & std::ios_base::iword (
    int __ix ) [inline]
```

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

precision() [1/2]

```
streamsize std::ios_base::precision ( ) const [inline]
```

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Referenced by [std::basic_ios<_CharT, _Traits>::copyfmt\(\)](#).

precision() [2/2]

```
streamsize std::ios_base::precision (
    streamsize __prec ) [inline]
```

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

pword()

```
void *& std::ios_base::pword (
    int __ix ) [inline]
```

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

register_callback()

```
void std::ios_base::register_callback (
    event_callback __fn,
    int __index )
```

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

setf() [1/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl ) [inline]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set. Referenced by [std::__detail::operator<>\(\)](#).

setf() [2/2]

```
fmtflags std::ios_base::setf (
    fmtflags __fmtfl,
    fmtflags __mask ) [inline]
```

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

sync_with_stdio()

```
static bool std::ios_base::sync_with_stdio (
    bool __sync = true ) [static]
```

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

unsetf()

```
void std::ios_base::unsetf (
    fmtflags __mask ) [inline]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

width() [1/2]

```
streamsize std::ios_base::width ( ) const [inline]
```

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, and `std::operator>>()`.

width() [2/2]

```
streamsize std::ios_base::width (
    streamsize __wide ) [inline]
```

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

xalloc()

```
static int std::ios_base::xalloc ( ) throw ( ) [static]
```

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

6.556.6 Member Data Documentation**adjustfield**

```
const fmtflags std::ios_base::adjustfield [static]
```

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Referenced by `std::num_put<_CharT, _OutIter>::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

app

```
const openmode std::ios_base::app [static]
```

Seek to end before each write.

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

ate

```
const openmode std::ios_base::ate [static]
```

Open and seek to end immediately after opening.

Referenced by [std::basic_filebuf< _CharT, _Traits >::open\(\)](#).

badbit

```
const iostate std::ios_base::badbit [static]
```

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Referenced by [std::basic_istream< _CharT, _Traits >::sentry::sentry\(\)](#), [std::basic_ios< _CharT, _Traits >::bad\(\)](#), [std::basic_ios< _CharT, _Traits >::fail\(\)](#), [std::basic_ostream< _CharT, _Traits >::flush\(\)](#), [std::basic_istream< _CharT, _Traits >::get\(\)](#), [std::basic_istream< char >::get\(\)](#), [std::basic_ostream< char, _Traits >::operator<<\(\)](#), [std::basic_ostream< _CharT, _Traits >::operator<<\(\)](#), [std::operator>>\(\)](#), [std::basic_istream< char >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#), [std::basic_istream< _CharT, _Traits >::peek\(\)](#), [std::basic_istream< char >::peek\(\)](#), [std::basic_ostream< _CharT, _Traits >::put\(\)](#), [std::basic_istream< _CharT, _Traits >::putback\(\)](#), [std::basic_istream< _CharT, _Traits >::readsome\(\)](#), [std::basic_istream< _CharT, _Traits >::seekg\(\)](#), [std::basic_istream< _CharT, _Traits >::sync\(\)](#), [std::basic_istream< _CharT, _Traits >::tellg\(\)](#), [std::basic_istream< _CharT, _Traits >::unget\(\)](#), and [std::ws\(\)](#).

basefield

```
const fmtflags std::ios_base::basefield [static]
```

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Referenced by [std::dec\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), [std::hex\(\)](#), and [std::oct\(\)](#).

beg

```
const seekdir std::ios_base::beg [static]
```

Request a seek relative to the beginning of the stream.

Referenced by [std::basic_filebuf< _CharT, _Traits >::seekpos\(\)](#).

binary

```
const openmode std::ios_base::binary [static]
```

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Referenced by [std::basic_filebuf< _CharT, _Traits >::showmanyc\(\)](#).

boolalpha

```
const fmtflags std::ios_base::boolalpha [static]
```

Insert/extract `bool` in alphabetic rather than numeric format.

Referenced by [std::boolalpha\(\)](#), [std::num_get< _CharT, _InIter >::do_get\(\)](#), [std::num_put< _CharT, _OutIter >::do_put\(\)](#), and [std::noboolalpha\(\)](#).

cur

```
const seekdir std::ios_base::cur [static]
```

Request a seek relative to the current position within the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_stringbuf<_CharT, _Traits, std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

dec

```
const fmtflags std::ios_base::dec [static]
```

Converts integer input or generates integer output in decimal base.

Referenced by `std::dec()`.

end

```
const seekdir std::ios_base::end [static]
```

Request a seek relative to the current end of the sequence.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

eofbit

```
const iostate std::ios_base::eofbit [static]
```

Indicates that an input operation reached the end of an input sequence.

```

Referenced by std::basic_istream<_CharT,_Traits>::sentry::sentry(), std::time_get<_CharT,_InIter>::do_get(),
std::num_get<_CharT,_InIter>::do_get(), std::num_get<_CharT,_InIter>::do_get(), std::num_get<_CharT,_InIter>::do_get(),
std::num_get<_CharT,_InIter>::do_get(), std::time_get<_CharT,_InIter>::do_get_date(), std::time_get<_CharT,_InIter>::do_get_
std::time_get<_CharT,_InIter>::do_get_time(), std::time_get<_CharT,_InIter>::do_get_weekday(), std::time_get<_CharT,_InIter>
std::basic_ios<_CharT,_Traits>::eof(), std::basic_istream<char>::get(), std::basic_istream<_CharT,_Traits>::get(),
std::basic_istream<_CharT,_Traits>::get(), std::time_get<_CharT,_InIter>::get(), std::basic_istream<_CharT,_Traits>::operator>
std::operator>>(), std::basic_istream<_CharT,_Traits>::peek(), std::basic_istream<_CharT,_Traits>::putback(),
std::basic_istream<_CharT,_Traits>::read(), std::basic_istream<_CharT,_Traits>::readsome(), std::basic_istream<char>::seekg()
std::basic_istream<_CharT,_Traits>::seekg(), std::basic_istream<_CharT,_Traits>::seekg(), std::basic_istream<_CharT,_Traits>:
and std::ws().

```

failbit

```
const iostate std::ios_base::failbit [static]
```

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<char>::ignore()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<char>::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

fixed

```
const fmtflags std::ios_base::fixed [static]
```

Generate floating-point output in fixed-point notation.

Referenced by `std::fixed()`, and `std::hexfloat()`.

floatfield

```
const fmtflags std::ios_base::floatfield [static]
```

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Referenced by [std::defaultfloat\(\)](#), [std::fixed\(\)](#), [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

goodbit

```
const iostate std::ios_base::goodbit [static]
```

Indicates all is well.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::time_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_monthname\(\)](#), [std::time_get<_CharT, _Inlter>::do_get_year\(\)](#), [std::basic_ostream<_CharT, _Traits>::flush\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::time_get<_CharT, _Inlter>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::basic_ostream<char, _Traits>::operator<<\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::operator>>\(\)](#), [std::basic_istream<char>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::operator>>\(\)](#), [std::basic_istream<_CharT, _Traits>::put\(\)](#), [std::basic_istream<_CharT, _Traits>::putback\(\)](#), [std::basic_istream<_CharT, _Traits>::readsome\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::sync\(\)](#), [std::basic_istream<_CharT, _Traits>::unset\(\)](#), and [std::ws\(\)](#).

hex

```
const fmtflags std::ios_base::hex [static]
```

Converts integer input or generates integer output in hexadecimal base.

Referenced by [std::num_get<_CharT, _Inlter>::do_get\(\)](#), [std::num_put<_CharT, _Outlter>::do_put\(\)](#), and [std::hex\(\)](#).

in

```
const openmode std::ios_base::in [static]
```

Open for input. Default for `ifstream` and `fstream`.

Referenced by [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::pbackfail\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_istream<_CharT, _Traits>::seekg\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::showmanyc\(\)](#), [std::basic_istream<_CharT, _Traits>::tellg\(\)](#), [std::basic_filebuf<_CharT, _Traits>::underflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::xsgetn\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsgetn\(\)](#).

internal

```
const fmtflags std::ios_base::internal [static]
```

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Referenced by [std::internal\(\)](#).

left

```
const fmtflags std::ios_base::left [static]
```

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Referenced by [std::num_put<_CharT, _Outlter>::do_put\(\)](#), and [std::left\(\)](#).

oct

```
const fmtflags std::ios_base::oct [static]
```

Converts integer input or generates integer output in octal base.
Referenced by [std::oct\(\)](#).

out

```
const openmode std::ios_base::out [static]
```

Open for output. Default for `ofstream` and `fstream`.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::pbackfail\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_ostream<_CharT, _Traits>::seekp\(\)](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>::tellp\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

right

```
const fmtflags std::ios_base::right [static]
```

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Referenced by [std::right\(\)](#).

scientific

```
const fmtflags std::ios_base::scientific [static]
```

Generates floating-point output in scientific notation.

Referenced by [std::hexfloat\(\)](#), and [std::scientific\(\)](#).

showbase

```
const fmtflags std::ios_base::showbase [static]
```

Generates a prefix indicating the numeric base of generated integer output.

Referenced by [std::num_put<_CharT, _Outiter>::do_put\(\)](#), [std::noshowbase\(\)](#), and [std::showbase\(\)](#).

showpoint

```
const fmtflags std::ios_base::showpoint [static]
```

Generates a decimal-point character unconditionally in generated floating-point output.

Referenced by [std::noshowpoint\(\)](#), and [std::showpoint\(\)](#).

showpos

```
const fmtflags std::ios_base::showpos [static]
```

Generates a + sign in non-negative generated numeric output.

Referenced by [std::noshowpos\(\)](#), and [std::showpos\(\)](#).

skipws

```
const fmtflags std::ios_base::skipws [static]
```

Skips leading white space before certain input operations.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::noskipws\(\)](#), [std::__detail::operator>>\(\)](#), and [std::skipws\(\)](#).

trunc

```
const openmode std::ios_base::trunc [static]
```

Truncate an existing stream when opening. Default for `ofstream`.

unitbuf

```
const fmt::ios_base::unitbuf [static]
```

Flushes output after each output operation.

Referenced by [std::nounitbuf\(\)](#), and [std::unitbuf\(\)](#).

uppercase

```
const fmt::ios_base::uppercase [static]
```

Replaces certain lowercase letters with their uppercase equivalents in generated output.

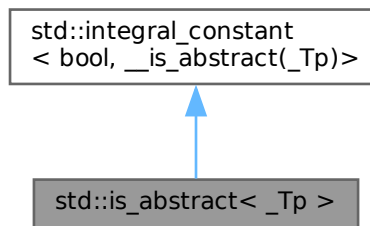
Referenced by [std::num_put<_CharT, _OutIter>::do_put\(\)](#), [std::nouppercase\(\)](#), and [std::uppercase\(\)](#).

The documentation for this class was generated from the following file:

- [ios_base.h](#)

6.557 std::is_abstract<_Tp> Struct Template Reference

Inheritance diagram for `std::is_abstract<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.557.1 Detailed Description

```
template<typename _Tp>  
struct std::is_abstract<_Tp>
```

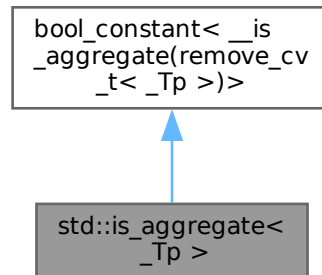
`is_abstract`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.558 std::is_aggregate< _Tp > Struct Template Reference

Inheritance diagram for std::is_aggregate< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.558.1 Detailed Description

```

template<typename _Tp>
struct std::is_aggregate< _Tp >

```

is_aggregate

Since

C++17

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.559 std::is_arithmetic< _Tp > Struct Template Reference

Inherits `__or_::type`.

6.559.1 Detailed Description

```
template<typename _Tp>
struct std::is_arithmetic<_Tp >
```

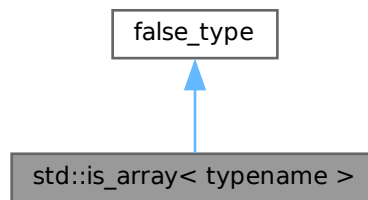
is_arithmetic

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.560 std::is_array< typename > Struct Template Reference

Inheritance diagram for std::is_array< typename >:



Public Types

- typedef [integral_constant](#)<_Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.560.1 Detailed Description

```
template<typename>
struct std::is_array< typename >
```

is_array

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.561 std::is_assignable< _Tp, _Up > Struct Template Reference

Inherits `__bool_constant< __is_assignable(_Tp, _Up)>`.

6.561.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_assignable< _Tp, _Up >
```

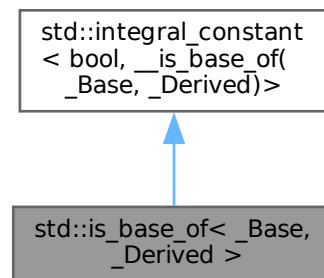
is_assignable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.562 std::is_base_of< _Base, _Derived > Struct Template Reference

Inheritance diagram for std::is_base_of< _Base, _Derived >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.562.1 Detailed Description

```
template<typename _Base, typename _Derived>
struct std::is_base_of< _Base, _Derived >
```

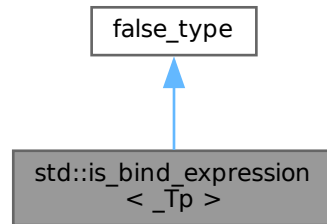
is_base_of

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.563 `std::is_bind_expression<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_bind_expression<_Tp>`:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.563.1 Detailed Description

```
template<typename _Tp>
struct std::is_bind_expression<_Tp>
```

Trait that identifies a bind expression.

Determines if the given type `_Tp` is a function object that should be treated as a subexpression when evaluating calls to function objects returned by `std::bind`.

C++11 [func.bind.isbind].

Since

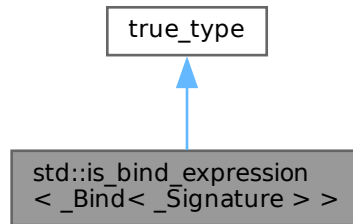
C++11

The documentation for this struct was generated from the following file:

- [functional](#)

6.564 std::is_bind_expression< _Bind< _Signature > > Struct Template Reference

Inheritance diagram for std::is_bind_expression< _Bind< _Signature > >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.564.1 Detailed Description

```
template<typename _Signature>  
struct std::is_bind_expression< _Bind< _Signature > >
```

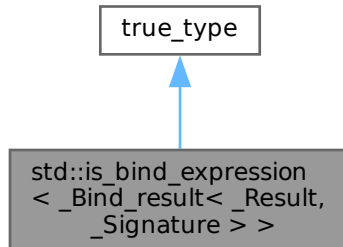
Class template _Bind is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.565 `std::is_bind_expression<_Bind_result<_Result,_Signature>>` Struct Template Reference

Inheritance diagram for `std::is_bind_expression<_Bind_result<_Result,_Signature>>`:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.565.1 Detailed Description

```
template<typename _Result, typename _Signature>
struct std::is_bind_expression<_Bind_result<_Result,_Signature>>>
```

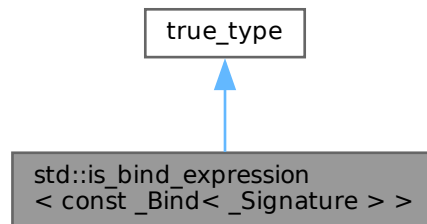
Class template `_Bind_result` is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.566 std::is_bind_expression< const _Bind< _Signature > > Struct Template Reference

Inheritance diagram for std::is_bind_expression< const _Bind< _Signature > >:

**Public Types**

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.566.1 Detailed Description

```

template<typename _Signature>
struct std::is_bind_expression< const _Bind< _Signature > >

```

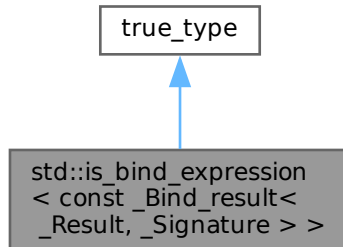
Class template _Bind is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.567 `std::is_bind_expression< const _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.567.1 Detailed Description

```
template<typename _Result, typename _Signature>  
struct std::is_bind_expression< const _Bind_result< _Result, _Signature > >
```

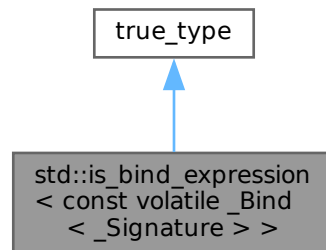
Class template `_Bind_result` is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.568 std::is_bind_expression< const volatile _Bind< _Signature > > Struct Template Reference

Inheritance diagram for std::is_bind_expression< const volatile _Bind< _Signature > >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.568.1 Detailed Description

template<typename _Signature>

struct std::is_bind_expression< const volatile _Bind< _Signature > >

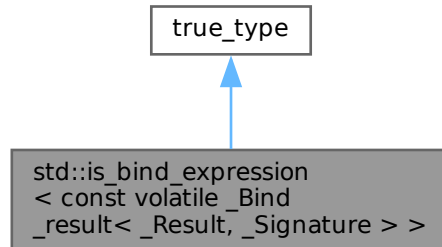
Class template _Bind is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.569 `std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.569.1 Detailed Description

```
template<typename _Result, typename _Signature>
struct std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >
```

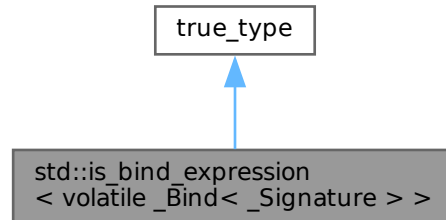
Class template `_Bind_result` is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.570 `std::is_bind_expression< volatile _Bind< _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< volatile _Bind< _Signature > >`:



Public Types

- typedef `integral_constant< _Tp, __v > type`
- typedef `_Tp value_type`

Public Member Functions

- constexpr `operator value_type ()` const noexcept
- constexpr `value_type operator() ()` const noexcept

Static Public Attributes

- static constexpr `_Tp value`

6.570.1 Detailed Description

```
template<typename _Signature>
struct std::is_bind_expression< volatile _Bind< _Signature > >
```

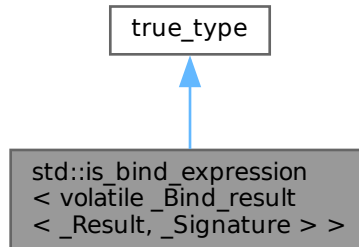
Class template `_Bind` is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.571 `std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.571.1 Detailed Description

```
template<typename _Result, typename _Signature>
struct std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >
```

Class template `_Bind_result` is always a bind expression.

The documentation for this struct was generated from the following file:

- [functional](#)

6.572 `std::is_bounded_array< _Tp >` Struct Template Reference

Inherits `__is_array_known_bounds< _Tp >`.

6.572.1 Detailed Description

```
template<typename _Tp>
struct std::is_bounded_array< _Tp >
```

True for a type that is an array of known bound.

Since

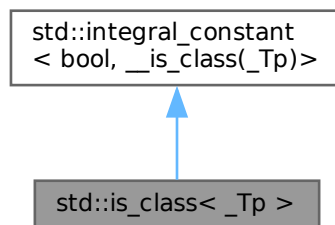
C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.573 std::is_class< _Tp > Struct Template Reference

Inheritance diagram for std::is_class< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.573.1 Detailed Description

```
template<typename _Tp>
struct std::is_class< _Tp >
```

is_class

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.574 std::is_compound< _Tp > Struct Template Reference

Inherits `__not_::type`.

6.574.1 Detailed Description

```
template<typename _Tp>
struct std::is_compound< _Tp >
```

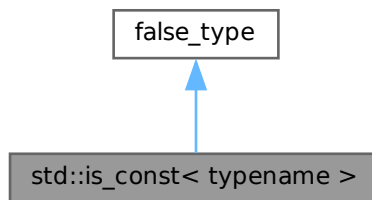
is_compound

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.575 std::is_const< typename > Struct Template Reference

Inheritance diagram for std::is_const< typename >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.575.1 Detailed Description

```
template<typename>
struct std::is_const< typename >
```

is_const

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.576 std::is_constructible< _Tp, _Args > Struct Template Reference

Inherits `__is_constructible_impl< _Tp, _Args... >`.

6.576.1 Detailed Description

```
template<typename _Tp, typename... _Args>
struct std::is_constructible<_Tp, _Args>
```

is_constructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.577 std::is_convertible<_From, _To> Struct Template Reference

Inherits __is_convertible_helper::type.

6.577.1 Detailed Description

```
template<typename _From, typename _To>
struct std::is_convertible<_From, _To>
```

is_convertible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.578 std::is_copy_assignable<_Tp> Struct Template Reference

Inherits __is_copy_assignable_impl::type.

6.578.1 Detailed Description

```
template<typename _Tp>
struct std::is_copy_assignable<_Tp>
```

is_copy_assignable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.579 std::is_copy_constructible<_Tp> Struct Template Reference

Inherits __is_copy_constructible_impl<_Tp>.

Inherited by std::__is_copy_insertable< allocator<_Tp> > >.

6.579.1 Detailed Description

```
template<typename _Tp>
struct std::is_copy_constructible<_Tp>
```

is_copy_constructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.580 std::is_default_constructible<_Tp> Struct Template Reference

Inherits __is_constructible_impl::type.

6.580.1 Detailed Description

```
template<typename _Tp>
struct std::is_default_constructible< _Tp >
```

is_default_constructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.581 std::is_destructible< _Tp > Struct Template Reference

Inherits `__is_destructible_safe::type`.

6.581.1 Detailed Description

```
template<typename _Tp>
struct std::is_destructible< _Tp >
```

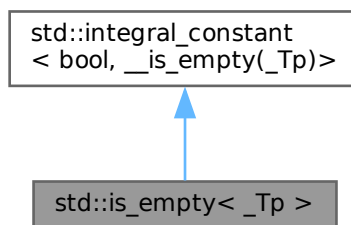
is_destructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.582 std::is_empty< _Tp > Struct Template Reference

Inheritance diagram for `std::is_empty< _Tp >`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.582.1 Detailed Description

```
template<typename _Tp>
struct std::is_empty< _Tp >
```

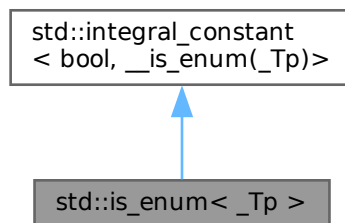
is_empty

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.583 std::is_enum< _Tp > Struct Template Reference

Inheritance diagram for std::is_enum< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.583.1 Detailed Description

```
template<typename _Tp>
struct std::is_enum< _Tp >
```

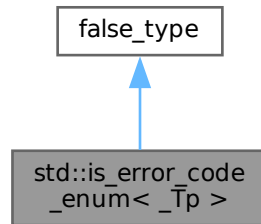
is_enum

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.584 std::is_error_code_enum< _Tp > Struct Template Reference

Inheritance diagram for std::is_error_code_enum< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.584.1 Detailed Description

```
template<typename _Tp>  
struct std::is_error_code_enum< _Tp >
```

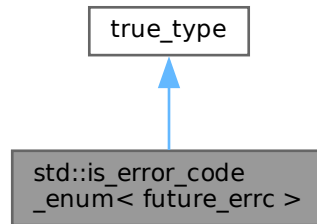
is_error_code_enum

The documentation for this struct was generated from the following file:

- [system_error](#)

6.585 std::is_error_code_enum< future_errc > Struct Reference

Inheritance diagram for std::is_error_code_enum< future_errc >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

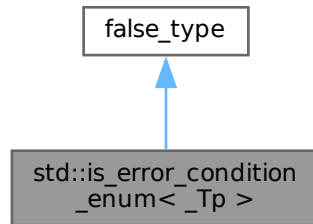
6.585.1 Detailed Description

Specialization that allows `future_errc` to convert to `error_code`.
The documentation for this struct was generated from the following file:

- [future](#)

6.586 std::is_error_condition_enum<_Tp> Struct Template Reference

Inheritance diagram for std::is_error_condition_enum<_Tp>:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.586.1 Detailed Description

```
template<typename _Tp>  
struct std::is_error_condition_enum<_Tp>
```

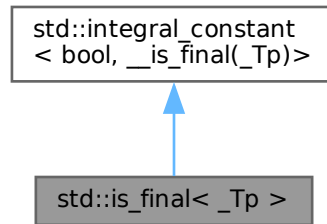
is_error_condition_enum

The documentation for this struct was generated from the following file:

- [system_error](#)

6.587 std::is_final< _Tp > Struct Template Reference

Inheritance diagram for std::is_final< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.587.1 Detailed Description

```
template<typename _Tp>
struct std::is_final< _Tp >
```

is_final

Since

C++14

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.588 std::is_floating_point< _Tp > Struct Template Reference

Inherits [__is_floating_point_helper::type](#).

6.588.1 Detailed Description

```
template<typename _Tp>
struct std::is_floating_point< _Tp >
```

is_floating_point

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.589 `std::is_function< _Tp >` Struct Template Reference

Inherits `__bool_constant<!is_const< const _Tp >::value >`.

6.589.1 Detailed Description

```
template<typename _Tp>
struct std::is_function< _Tp >
```

`is_function`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.590 `std::is_fundamental< _Tp >` Struct Template Reference

Inherits `__or_::type`.

6.590.1 Detailed Description

```
template<typename _Tp>
struct std::is_fundamental< _Tp >
```

`is_fundamental`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.591 `std::is_integral< _Tp >` Struct Template Reference

Inherits `__is_integral_helper::type`.

6.591.1 Detailed Description

```
template<typename _Tp>
struct std::is_integral< _Tp >
```

`is_integral`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.592 `std::is_invocable< _Fn, _ArgTypes >` Struct Template Reference

Inherits `__is_invocable_impl::type`.

6.592.1 Detailed Description

```
template<typename _Fn, typename... _ArgTypes>
struct std::is_invocable< _Fn, _ArgTypes >
```

`std::is_invocable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.593 std::is_invocable_r< _Ret, _Fn, _ArgTypes > Struct Template Reference

Inherits `__is_invocable_impl::type`.

6.593.1 Detailed Description

```
template<typename _Ret, typename _Fn, typename... _ArgTypes>
struct std::is_invocable_r< _Ret, _Fn, _ArgTypes >
```

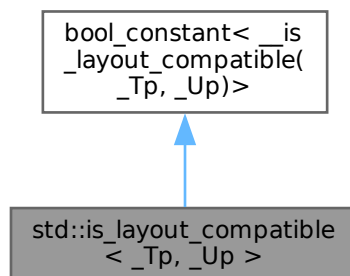
`std::is_invocable_r`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.594 std::is_layout_compatible< _Tp, _Up > Struct Template Reference

Inheritance diagram for `std::is_layout_compatible< _Tp, _Up >`:

**Public Types**

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.594.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_layout_compatible< _Tp, _Up >
```

Since

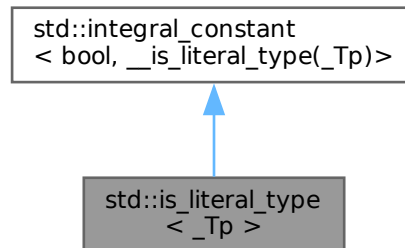
C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.595 std::is_literal_type<_Tp> Struct Template Reference

Inheritance diagram for std::is_literal_type<_Tp>:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.595.1 Detailed Description

```
template<typename _Tp>
struct std::is_literal_type<_Tp>
```

is_literal_type

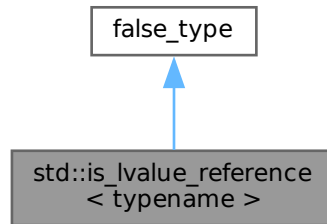
Deprecated Deprecated in C++17, removed in C++20. The idea of a literal type isn't useful.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.596 std::is_lvalue_reference< typename > Struct Template Reference

Inheritance diagram for std::is_lvalue_reference< typename >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.596.1 Detailed Description

```
template<typename>  
struct std::is_lvalue_reference< typename >
```

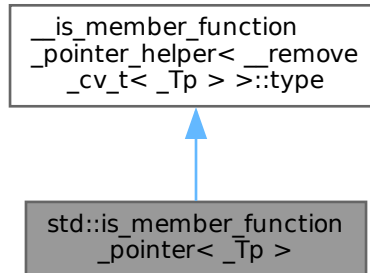
is_lvalue_reference

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.597 std::is_member_function_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_member_function_pointer< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.597.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_function_pointer< _Tp >
```

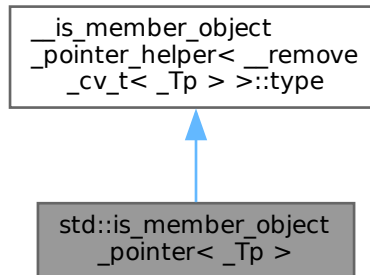
is_member_function_pointer

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.598 std::is_member_object_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_member_object_pointer< _Tp >:

**Public Types**

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.598.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_object_pointer< _Tp >
```

is_member_object_pointer

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.599 std::is_member_pointer< _Tp > Struct Template Reference

Inherits __is_member_pointer_helper::type.

6.599.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_pointer< _Tp >
```

is_member_pointer

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.600 `std::is_move_assignable<_Tp>` Struct Template Reference

Inherits `__is_move_assignable_impl::type`.

6.600.1 Detailed Description

```
template<typename _Tp>
struct std::is_move_assignable<_Tp>
```

`is_move_assignable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.601 `std::is_move_constructible<_Tp>` Struct Template Reference

Inherits `__is_move_constructible_impl<_Tp>`.

Inherited by `std::__is_move_insertable< allocator<_Tp>>`.

6.601.1 Detailed Description

```
template<typename _Tp>
struct std::is_move_constructible<_Tp>
```

`is_move_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.602 `std::is_nothrow_assignable<_Tp, _Up>` Struct Template Reference

Inherits `__is_nothrow_assignable_impl<_Tp, _Up>`.

6.602.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_nothrow_assignable<_Tp, _Up>
```

`is_nothrow_assignable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.603 `std::is_nothrow_constructible<_Tp, _Args>` Struct Template Reference

Inherits `__is_nothrow_constructible_impl::type`.

6.603.1 Detailed Description

```
template<typename _Tp, typename... _Args>
struct std::is_nothrow_constructible<_Tp, _Args>
```

`is_nothrow_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.604 `std::is_nothrow_convertible<_From, _To>` Struct Template Reference

Inherits `__is_nt_convertible_helper::type`.

6.604.1 Detailed Description

```
template<typename _From, typename _To>
struct std::is_nothrow_convertible<_From, _To>
```

`is_nothrow_convertible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.605 `std::is_nothrow_copy_assignable<_Tp>` Struct Template Reference

Inherits `std::__is_nt_copy_assignable_impl<_Tp, bool>`.

6.605.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_copy_assignable<_Tp>
```

`is_nothrow_copy_assignable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.606 `std::is_nothrow_copy_constructible<_Tp>` Struct Template Reference

Inherits `__is_nothrow_copy_constructible_impl::type`.

6.606.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_copy_constructible<_Tp>
```

`is_nothrow_copy_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.607 `std::is_nothrow_default_constructible<_Tp>` Struct Template Reference

Inherits `__bool_constant<__is_nothrow_constructible(_Tp)>`.

6.607.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_default_constructible<_Tp>
```

`is_nothrow_default_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.608 `std::is_nothrow_destructible<_Tp>` Struct Template Reference

Inherits `__is_nt_destructible_safe::type`.

6.608.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_destructible< _Tp >
```

is_nothrow_destructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.609 std::is_nothrow_invocable< _Fn, _ArgTypes > Struct Template Reference

Inherits `__and_::type`.

6.609.1 Detailed Description

```
template<typename _Fn, typename... _ArgTypes>
struct std::is_nothrow_invocable< _Fn, _ArgTypes >
```

std::is_nothrow_invocable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.610 std::is_nothrow_invocable_r< _Ret, _Fn, _ArgTypes > Struct Template Reference

Inherits `__and_::type`.

6.610.1 Detailed Description

```
template<typename _Ret, typename _Fn, typename... _ArgTypes>
struct std::is_nothrow_invocable_r< _Ret, _Fn, _ArgTypes >
```

std::is_nothrow_invocable_r

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.611 std::is_nothrow_move_assignable< _Tp > Struct Template Reference

Inherits `std::__is_nt_move_assignable_impl< _Tp, bool >`.

6.611.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_move_assignable< _Tp >
```

is_nothrow_move_assignable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.612 std::is_nothrow_move_constructible< _Tp > Struct Template Reference

Inherits `__is_nothrow_move_constructible_impl::type`.

6.612.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_move_constructible< _Tp >
```

`is_nothrow_move_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.613 `std::is_nothrow_swappable< _Tp >` Struct Template Reference

Inherits `__is_nothrow_swappable_impl::type`.

6.613.1 Detailed Description

```
template<typename _Tp>
struct std::is_nothrow_swappable< _Tp >
```

`is_nothrow_swappable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.614 `std::is_nothrow_swappable_with< _Tp, _Up >` Struct Template Reference

Inherits `__is_nothrow_swappable_with_impl::type`.

6.614.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_nothrow_swappable_with< _Tp, _Up >
```

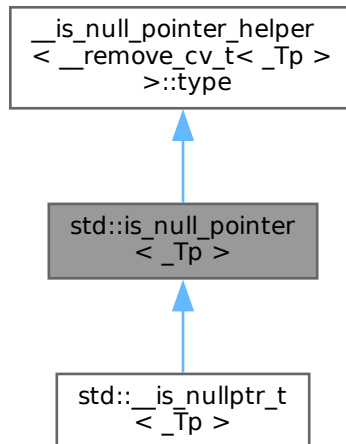
`is_nothrow_swappable_with`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.615 std::is_null_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_null_pointer< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.615.1 Detailed Description

template<typename _Tp>
struct std::is_null_pointer< _Tp >

is_null_pointer (LWG 2247).

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.616 std::is_object< _Tp > Struct Template Reference

Inherits `__not_::type`.

6.616.1 Detailed Description

```
template<typename _Tp>
struct std::is_object<_Tp>
```

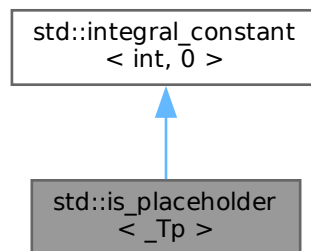
is_object

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.617 std::is_placeholder<_Tp> Struct Template Reference

Inheritance diagram for std::is_placeholder<_Tp>:



Public Types

- typedef [integral_constant](#)< int, __v > **type**
- typedef int **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr int **value**

6.617.1 Detailed Description

```
template<typename _Tp>
struct std::is_placeholder<_Tp>
```

Determines if the given type _Tp is a placeholder in a bind() expression and, if so, which placeholder it is. C++11 [func.bind.isplace].

Since

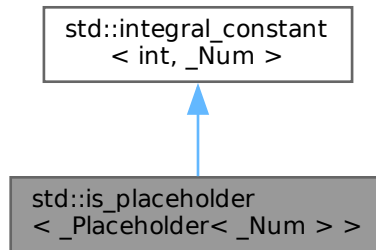
C++11

The documentation for this struct was generated from the following file:

- [functional](#)

6.618 std::is_placeholder<_Placeholder<_Num>> Struct Template Reference

Inheritance diagram for std::is_placeholder<_Placeholder<_Num>>:



Public Types

- typedef `integral_constant< int, __v >` **type**
- typedef int **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr int **value**

6.618.1 Detailed Description

```
template<int _Num>
```

```
struct std::is_placeholder<_Placeholder<_Num>>
```

Partial specialization of `is_placeholder` that provides the placeholder number for the placeholder objects defined by `libstdc++`.

Since

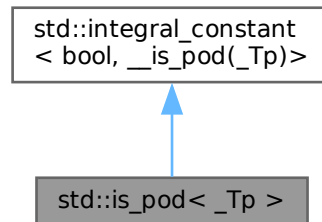
C++11

The documentation for this struct was generated from the following file:

- [functional](#)

6.619 std::is_pod< _Tp > Struct Template Reference

Inheritance diagram for std::is_pod< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.619.1 Detailed Description

```
template<typename _Tp>
struct std::is_pod< _Tp >
```

is_pod

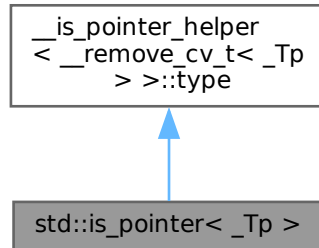
Deprecated Deprecated in C++20. Use `is_standard_layout` && `is_trivial` instead.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.620 std::is_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_pointer< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.620.1 Detailed Description

```
template<typename _Tp>
struct std::is_pointer< _Tp >
```

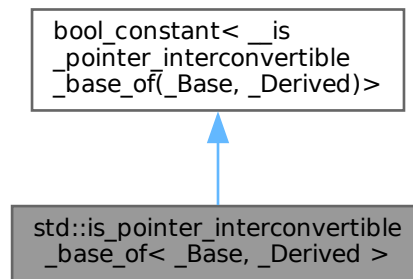
is_pointer

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.621 `std::is_pointer_interconvertible_base_of<_Base, _Derived>` Struct Template Reference

Inheritance diagram for `std::is_pointer_interconvertible_base_of<_Base, _Derived>`:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.621.1 Detailed Description

```
template<typename _Base, typename _Derived>
struct std::is_pointer_interconvertible_base_of<_Base, _Derived>
```

True if `_Derived` is standard-layout and has a base class of type `_Base`

Since

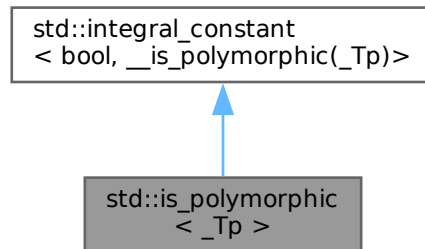
C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.622 `std::is_polymorphic<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_polymorphic<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.622.1 Detailed Description

```
template<typename _Tp>
struct std::is_polymorphic<_Tp>
```

`is_polymorphic`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.623 `std::is_reference<_Tp>` Struct Template Reference

Inherits `__or_::type`.

6.623.1 Detailed Description

```
template<typename _Tp>
struct std::is_reference<_Tp>
```

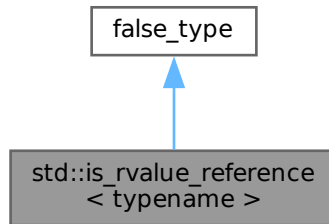
`is_reference`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.624 std::is_rvalue_reference< typename > Struct Template Reference

Inheritance diagram for std::is_rvalue_reference< typename >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr _Tp **value**

6.624.1 Detailed Description

```
template<typename>  
struct std::is_rvalue_reference< typename >
```

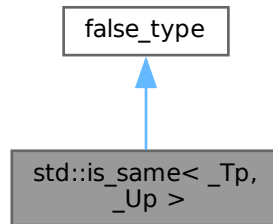
is_rvalue_reference

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.625 `std::is_same< _Tp, _Up >` Struct Template Reference

Inheritance diagram for `std::is_same< _Tp, _Up >`:



Public Types

- typedef [integral_constant< _Tp, __v >](#) **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `_Tp` **value**

6.625.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_same< _Tp, _Up >
```

`is_same`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.626 `std::is_scalar< _Tp >` Struct Template Reference

Inherits `__or_::type`.

6.626.1 Detailed Description

```
template<typename _Tp>
struct std::is_scalar< _Tp >
```

`is_scalar`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.627 std::is_signed< _Tp > Struct Template Reference

Inherits `__is_signed_helper::type`.

6.627.1 Detailed Description

```
template<typename _Tp>
struct std::is_signed< _Tp >
```

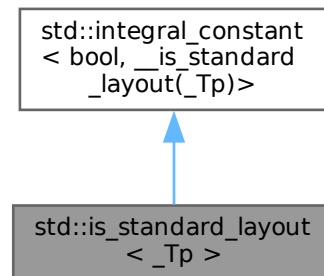
`is_signed`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.628 std::is_standard_layout< _Tp > Struct Template Reference

Inheritance diagram for `std::is_standard_layout< _Tp >`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.628.1 Detailed Description

```
template<typename _Tp>
struct std::is_standard_layout< _Tp >
```

`is_standard_layout`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.629 `std::is_swappable< _Tp >` Struct Template Reference

Inherits `__is_swappable_impl::type`.

6.629.1 Detailed Description

```
template<typename _Tp>
struct std::is_swappable< _Tp >
```

Metafunctions used for detecting swappable types: p0185r1.

`is_swappable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.630 `std::is_swappable_with< _Tp, _Up >` Struct Template Reference

Inherits `__is_swappable_with_impl::type`.

6.630.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_swappable_with< _Tp, _Up >
```

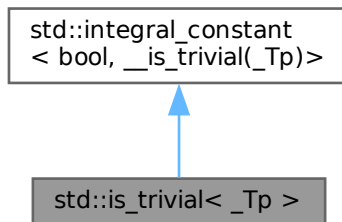
`is_swappable_with`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.631 `std::is_trivial< _Tp >` Struct Template Reference

Inheritance diagram for `std::is_trivial< _Tp >`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.631.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivial< _Tp >
```

is_trivial

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.632 std::is_trivially_assignable< _Tp, _Up > Struct Template Reference

Inherits `__bool_constant< __is_trivially_assignable(_Tp, _Up)>`.

6.632.1 Detailed Description

```
template<typename _Tp, typename _Up>
struct std::is_trivially_assignable< _Tp, _Up >
```

is_trivially_assignable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.633 std::is_trivially_constructible< _Tp, _Args > Struct Template Reference

Inherits `__bool_constant< __is_trivially_constructible(_Tp, _Args...)>`.

6.633.1 Detailed Description

```
template<typename _Tp, typename... _Args>
struct std::is_trivially_constructible< _Tp, _Args >
```

is_trivially_constructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.634 std::is_trivially_copy_assignable< _Tp > Struct Template Reference

Inherits `std::__is_trivially_copy_assignable_impl< _Tp, bool >`.

6.634.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_copy_assignable< _Tp >
```

is_trivially_copy_assignable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.635 std::is_trivially_copy_constructible< _Tp > Struct Template Reference

Inherits `std::__is_trivially_copy_constructible_impl< _Tp, bool >`.

6.635.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_copy_constructible< _Tp >
```

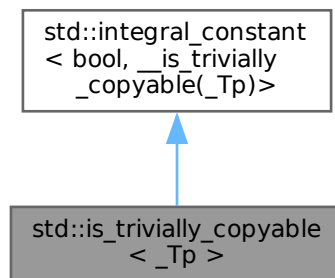
is_trivially_copy_constructible

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.636 std::is_trivially_copyable< _Tp > Struct Template Reference

Inheritance diagram for std::is_trivially_copyable< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.636.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_copyable< _Tp >
```

is_trivially_copyable

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.637 `std::is_trivially_default_constructible<_Tp>` Struct Template Reference

Inherits `__bool_constant<__is_trivially_constructible(_Tp)>`.

6.637.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_default_constructible<_Tp>
```

`is_trivially_default_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.638 `std::is_trivially_destructible<_Tp>` Struct Template Reference

Inherits `__and_<__is_destructible_safe<_Tp>, __bool_constant<__has_trivial_destructor(_Tp)>>`.

6.638.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_destructible<_Tp>
```

`is_trivially_destructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.639 `std::is_trivially_move_assignable<_Tp>` Struct Template Reference

Inherits `std::__is_trivially_move_assignable_impl<_Tp, bool>`.

6.639.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_move_assignable<_Tp>
```

`is_trivially_move_assignable`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.640 `std::is_trivially_move_constructible<_Tp>` Struct Template Reference

Inherits `std::__is_trivially_move_constructible_impl<_Tp, bool>`.

6.640.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivially_move_constructible<_Tp>
```

`is_trivially_move_constructible`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.641 `std::is_unbounded_array<_Tp>` Struct Template Reference

Inherits `__is_array_unknown_bounds<_Tp>`.

6.641.1 Detailed Description

```
template<typename _Tp>
struct std::is_unbounded_array< _Tp >
```

True for a type that is an array of unknown bound.

Since

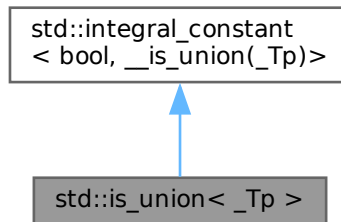
C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.642 std::is_union< _Tp > Struct Template Reference

Inheritance diagram for std::is_union< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.642.1 Detailed Description

```
template<typename _Tp>
struct std::is_union< _Tp >
```

is_union

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.643 std::is_unsigned< _Tp > Struct Template Reference

Inherits `__and_< is_arithmetic< _Tp >, __not_< is_signed< _Tp > > >`.

6.643.1 Detailed Description

```
template<typename _Tp>
struct std::is_unsigned< _Tp >
```

`is_unsigned`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.644 std::is_void< _Tp > Struct Template Reference

Inherits `__is_void_helper::type`.

6.644.1 Detailed Description

```
template<typename _Tp>
struct std::is_void< _Tp >
```

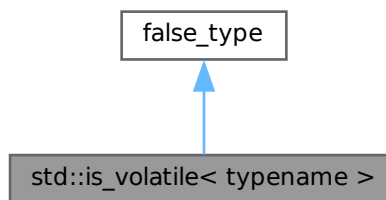
`is_void`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.645 std::is_volatile< typename > Struct Template Reference

Inheritance diagram for `std::is_volatile< typename >`:



Public Types

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `_Tp` **value**

6.645.1 Detailed Description

template<typename>
struct `std::is_volatile`< typename >

`is_volatile`

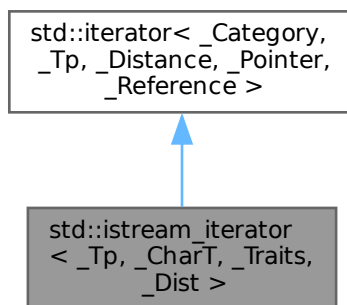
The documentation for this struct was generated from the following file:

- [type_traits](#)

6.646 `std::istream_iterator`< `_Tp`, `_CharT`, `_Traits`, `_Dist` > Class Template Reference

`#include <stream_iterator.h>`

Inheritance diagram for `std::istream_iterator`< `_Tp`, `_CharT`, `_Traits`, `_Dist` >:



Public Types

- typedef `_CharT` **char_type**
- typedef `_Distance` [difference_type](#)
- typedef [basic_istream](#)< `_CharT`, `_Traits` > **istream_type**
- typedef `_Category` [iterator_category](#)
- typedef `_Pointer` [pointer](#)
- typedef `_Reference` [reference](#)
- typedef `_Traits` **traits_type**
- typedef `_Tp` [value_type](#)

Public Member Functions

- constexpr [istream_iterator](#) () noexcept(*/*conditional */*)
- **istream_iterator** (const [istream_iterator](#) &__obj) noexcept(*/*conditional */*)
- [istream_iterator](#) ([istream_type](#) &__s)
- const `_Tp` & **operator*** () const noexcept
- [istream_iterator](#) & **operator++** ()

- [istream_iterator](#) **operator++** (int)
- const _Tp * **operator->** () const noexcept
- [istream_iterator](#) & **operator=** (const [istream_iterator](#) &)=default

Friends

- bool [operator==](#) (const [istream_iterator](#) &__x, const [istream_iterator](#) &__y) noexcept

6.646.1 Detailed Description

template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>, typename _Dist = ptrdiff_t>

class std::istream_iterator< _Tp, _CharT, _Traits, _Dist >

Provides input iterator semantics for streams.

6.646.2 Member Typedef Documentation

difference_type

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Distance std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::difference_↵
type [inherited]
```

Distance between iterators is represented as this type.

iterator_category

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Category std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::iterator_↵
category [inherited]
```

One of the [tag types](#).

pointer

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Pointer std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::pointer [inherited]
```

This type represents a pointer-to-value_type.

reference

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Reference std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::reference
[inherited]
```

This type represents a reference-to-value_type.

value_type

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Tp std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::value_type [inherited]
```

The type "pointed to" by the iterator.

6.646.3 Constructor & Destructor Documentation

istream_iterator() [1/2]

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>, typename
_Dist = ptrdiff_t>
```

```
constexpr std::istream\_iterator< _Tp, _CharT, _Traits, _Dist >::istream_iterator ( ) [inline],
[constexpr], [noexcept]
```

Construct end of input stream iterator.

istream_iterator() [2/2]

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>, typename
_Dist = ptrdiff_t>
```

```
std::istream\_iterator< _Tp, _CharT, _Traits, _Dist >::istream_iterator (
    istream\_type & __s ) [inline]
```

Construct start of input stream iterator.

6.646.4 Friends And Related Symbol Documentation

operator==

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>, typename
_Dist = ptrdiff_t>
```

```
bool operator== (
    const istream\_iterator< _Tp, _CharT, _Traits, _Dist > & __x,
    const istream\_iterator< _Tp, _CharT, _Traits, _Dist > & __y ) [friend]
```

Return true if the iterators refer to the same stream, or are both at end-of-stream.

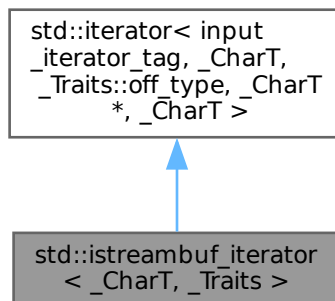
The documentation for this class was generated from the following file:

- [stream_iterator.h](#)

6.647 std::istreambuf_iterator< _CharT, _Traits > Class Template Reference

```
#include <streambuf_iterator.h>
```

Inheritance diagram for `std::istreambuf_iterator< _CharT, _Traits >`:



Public Types

- typedef `_Traits::off_type` [difference_type](#)
- typedef [input_iterator_tag](#) `iterator_category`
- typedef `_CharT` [reference](#)
- typedef `_CharT` [value_type](#)
- using [pointer](#) = void
- typedef `_CharT` [char_type](#)
- typedef `_Traits` [traits_type](#)
- typedef `_Traits::int_type` [int_type](#)
- typedef [basic_streambuf](#)< `_CharT`, `_Traits` > [streambuf_type](#)
- typedef [basic_istream](#)< `_CharT`, `_Traits` > [istream_type](#)

Public Member Functions

- constexpr [istreambuf_iterator](#) () noexcept
- [istreambuf_iterator](#) (const [istreambuf_iterator](#) &) noexcept=default
- constexpr [istreambuf_iterator](#) ([default_sentinel_t](#)) noexcept
- [istreambuf_iterator](#) ([istream_type](#) &__s) noexcept
- [istreambuf_iterator](#) ([streambuf_type](#) *__s) noexcept
- bool [equal](#) (const [istreambuf_iterator](#) &__b) const
- [char_type](#) [operator*](#) () const
- [istreambuf_iterator](#) & [operator++](#) ()
- [istreambuf_iterator](#) [operator++](#) (int)
- [istreambuf_iterator](#) & [operator=](#) (const [istreambuf_iterator](#) &) noexcept=default

Friends

- template<bool `_IsMove`, typename `_CharT2` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, _CharT2 * >::__type` [__copy_move_a2](#)
([istreambuf_iterator](#)< `_CharT2` >, [istreambuf_iterator](#)< `_CharT2` >, `_CharT2 *`)
- template<typename `_CharT2`, typename `_Size` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, _CharT2 * >::__type` [__copy_n_a](#) ([istreambuf_iterator](#)<
`_CharT2` >, `_Size`, `_CharT2 *`, bool)
- template<typename `_CharT2`, typename `_Distance` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, void >::__type` [advance](#) ([istreambuf_iterator](#)< `_CharT2` > &, `_Distance`)
- template<typename `_CharT2` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, ostreambuf_iterator< _CharT2 >::__type` [copy](#)
([istreambuf_iterator](#)< `_CharT2` >, [istreambuf_iterator](#)< `_CharT2` >, [ostreambuf_iterator](#)< `_CharT2` >)
- template<typename `_CharT2` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, istreambuf_iterator< _CharT2 >::__type` [find](#)
([istreambuf_iterator](#)< `_CharT2` >, [istreambuf_iterator](#)< `_CharT2` >, const `_CharT2` &)
- bool [operator==](#) (const [istreambuf_iterator](#) &__i, [default_sentinel_t](#) __s)

6.647.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::istreambuf_iterator< _CharT, _Traits >
```

Provides input iterator semantics for streambufs.

6.647.2 Member Typedef Documentation

char_type

```
template<typename _CharT , typename _Traits >
typedef _CharT std::istreambuf_iterator< _CharT, _Traits >::char_type
Public typedefs.
```

difference_type

```
typedef _Traits::off_type std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _CharT
* , _CharT >::difference_type [inherited]
Distance between iterators is represented as this type.
```

int_type

```
template<typename _CharT , typename _Traits >
typedef _Traits::int_type std::istreambuf_iterator< _CharT, _Traits >::int_type
Public typedefs.
```

istream_type

```
template<typename _CharT , typename _Traits >
typedef basic_istream<_CharT, _Traits> std::istreambuf_iterator< _CharT, _Traits >::istream_type
Public typedefs.
```

iterator_category

```
typedef input_iterator_tag std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _↵
CharT * , _CharT >::iterator_category [inherited]
One of the tag types.
```

pointer

```
template<typename _CharT , typename _Traits >
using std::istreambuf_iterator< _CharT, _Traits >::pointer = void
Public typedefs.
```

reference

```
typedef _CharT std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _CharT * , _CharT
>::reference [inherited]
This type represents a reference-to-value_type.
```

streambuf_type

```
template<typename _CharT , typename _Traits >
typedef basic_streambuf<_CharT, _Traits> std::istreambuf_iterator< _CharT, _Traits >::streambuf↵
_type
Public typedefs.
```

traits_type

```
template<typename _CharT , typename _Traits >
typedef _Traits std::istreambuf_iterator< _CharT, _Traits >::traits_type
Public typedefs.
```

value_type

```
typedef _CharT std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _CharT * , _CharT
>::value_type [inherited]
```

The type "pointed to" by the iterator.

6.647.3 Constructor & Destructor Documentation**istreambuf_iterator() [1/3]**

```
template<typename _CharT , typename _Traits >
constexpr std::istreambuf_iterator< _CharT, _Traits >::istreambuf_iterator ( ) [inline], [constexpr],
[noexcept]
```

Construct end of input stream iterator.

istreambuf_iterator() [2/3]

```
template<typename _CharT , typename _Traits >
std::istreambuf_iterator< _CharT, _Traits >::istreambuf_iterator (
    istream_type & __s ) [inline], [noexcept]
```

Construct start of input stream iterator.

istreambuf_iterator() [3/3]

```
template<typename _CharT , typename _Traits >
std::istreambuf_iterator< _CharT, _Traits >::istreambuf_iterator (
    streambuf_type * __s ) [inline], [noexcept]
```

Construct start of streambuf iterator.

6.647.4 Member Function Documentation**equal()**

```
template<typename _CharT , typename _Traits >
bool std::istreambuf_iterator< _CharT, _Traits >::equal (
    const istreambuf_iterator< _CharT, _Traits > & __b ) const [inline]
```

Return true both iterators are end or both are not end.

operator*()

```
template<typename _CharT , typename _Traits >
char_type std::istreambuf_iterator< _CharT, _Traits >::operator* ( ) const [inline]
```

Return the current character pointed to by iterator. This returns streambuf.sgetc(). It cannot be assigned. NB: The result of operator*() on an end of stream is undefined.

operator++() [1/2]

```
template<typename _CharT , typename _Traits >
istreambuf_iterator & std::istreambuf_iterator< _CharT, _Traits >::operator++ ( ) [inline]
```

Advance the iterator. Calls streambuf.sbumpc().

References [std::basic_streambuf< _CharT, _Traits >::sbumpc\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::sgetc\(\)](#).

operator++() [2/2]

```
template<typename _CharT , typename _Traits >
istreambuf_iterator std::istreambuf_iterator< _CharT, _Traits >::operator++ (
    int ) [inline]
```

Advance the iterator. Calls `streambuf.sbumpc()`.

References [std::basic_streambuf<_CharT, _Traits>::sbumpc\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#).

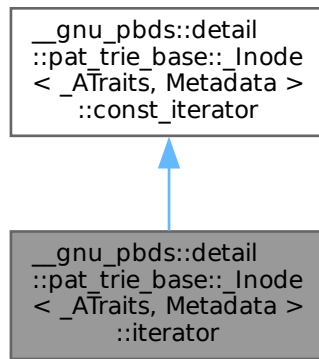
The documentation for this class was generated from the following files:

- [iosfwd](#)
- [streambuf_iterator.h](#)

6.648 `__gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>::iterator` Struct Reference

`#include <pat_trie_base.hpp>`

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata>::iterator`:



Public Types

- typedef `_Alloc::difference_type` **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef `node_pointer_pointer` **pointer**
- typedef `node_pointer_reference` **reference**
- typedef `node_pointer` **value_type**

Public Member Functions

- **iterator** (`node_pointer_pointer p_p_cur=0, node_pointer_pointer p_p_end=0`)
- **operator!=** (`const const_iterator &other`) `const`
- **operator!=** (`const iterator &other`) `const`
- `node_pointer` **operator*** ()
- `node_const_pointer` **operator*** () `const`
- [iterator](#) & **operator++** ()
- [iterator](#) **operator++** (`int`)
- `node_pointer_pointer` **operator->** ()
- `const node_pointer_pointer` **operator->** () `const`
- **operator==** (`const const_iterator &other`) `const`
- **operator==** (`const iterator &other`) `const`

Public Attributes

- node_pointer_pointer **m_p_p_cur**
- node_pointer_pointer **m_p_p_end**

6.648.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::iterator
```

Child iterator.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.649 std::experimental::filesystem::v1::path::iterator Class Reference

```
#include <fs_path.h>
```

Public Types

- using **difference_type** = std::ptrdiff_t
- using **iterator_category** = [std::bidirectional_iterator_tag](#)
- using **pointer** = const [path](#) *
- using **reference** = const [path](#) &
- using **value_type** = [path](#)

Public Member Functions

- **iterator** (const [iterator](#) &)=default
- **reference operator*** () const noexcept
- **iterator & operator++** () noexcept
- **iterator operator++** (int) noexcept
- **iterator & operator--** () noexcept
- **iterator operator--** (int) noexcept
- **pointer operator->** () const noexcept
- **iterator & operator=** (const [iterator](#) &)=default

Friends

- bool **operator!=** (const [iterator](#) &__lhs, const [iterator](#) &__rhs) noexcept
- bool **operator==** (const [iterator](#) &__lhs, const [iterator](#) &__rhs) noexcept
- class **path**

6.649.1 Detailed Description

An iterator for the components of a path.

The documentation for this class was generated from the following file:

- [experimental/bits/fs_path.h](#)

6.650 std::filesystem::path::iterator Class Reference

```
#include <fs_path.h>
```

Public Types

- using **difference_type** = std::ptrdiff_t
- using **iterator_category** = [std::bidirectional_iterator_tag](#)
- using **pointer** = const [path](#) *
- using **reference** = const [path](#) &
- using **value_type** = [path](#)

Public Member Functions

- **iterator** (const [iterator](#) &)=default
- **reference operator*** () const noexcept
- **iterator & operator++** () noexcept
- **iterator operator++** (int) noexcept
- **iterator & operator--** () noexcept
- **iterator operator--** (int) noexcept
- **pointer operator->** () const noexcept
- **iterator & operator=** (const [iterator](#) &)=default

Friends

- void **__path_iter_advance** ([iterator](#) &__i, difference_type __n) noexcept
- difference_type **__path_iter_distance** (const [iterator](#) &__first, const [iterator](#) &__last) noexcept
- bool **operator!=** (const [iterator](#) &__lhs, const [iterator](#) &__rhs) noexcept
- bool **operator==** (const [iterator](#) &__lhs, const [iterator](#) &__rhs) noexcept
- class **path**

6.650.1 Detailed Description

An iterator for the components of a path.

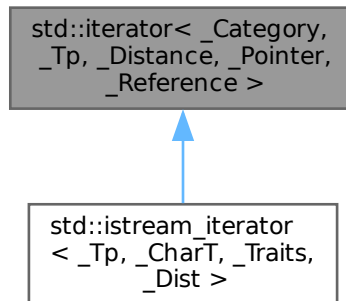
The documentation for this class was generated from the following file:

- [bits/fs_path.h](#)

6.651 `std::iterator<_Category, _Tp, _Distance, _Pointer, _Reference >` Struct Template Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`:



Public Types

- typedef `_Distance` [difference_type](#)
- typedef `_Category` [iterator_category](#)
- typedef `_Pointer` [pointer](#)
- typedef `_Reference` [reference](#)
- typedef `_Tp` [value_type](#)

6.651.1 Detailed Description

template<typename `_Category`, typename `_Tp`, typename `_Distance` = `ptrdiff_t`, typename `_Pointer` = `_Tp*`, typename `_Reference` = `_Tp&`>

struct `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`

Common iterator class.

This class does nothing but define nested typedefs. Iterator classes can inherit from this class to save some work. The typedefs are then used in specializations and overloading.

In particular, there are no default implementations of requirements such as `operator++` and the like. (How could there be?)

6.651.2 Member Typedef Documentation

`difference_type`

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Distance std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::difference_↵
type
```

Distance between iterators is represented as this type.

`iterator_category`

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
```



```
typedef _Category std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::iterator_↵
category
One of the tag types.
```

pointer

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Pointer std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::pointer
This type represents a pointer-to-value_type.
```

reference

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Reference std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::reference
This type represents a reference-to-value_type.
```

value_type

```
template<typename _Category , typename _Tp , typename _Distance = ptrdiff_t, typename _Pointer =
_Tp*, typename _Reference = _Tp&>
typedef _Tp std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::value_type
The type "pointed to" by the iterator.
```

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.652 std::iterator_traits< _Iterator > Struct Template Reference

Inherits `std::__iterator_traits< _Iterator, typename >`.

6.652.1 Detailed Description

```
template<typename _Iterator>
struct std::iterator_traits< _Iterator >
```

Traits class for iterators.

This class does nothing but define nested typedefs. The general version simply *forwards* the nested typedefs from the `Iterator` argument. Specialized versions for pointers and pointers-to-const provide tighter, more correct semantics.

The documentation for this struct was generated from the following file:

- [cpp_type_traits.h](#)

6.653 std::iterator_traits< _Tp * > Struct Template Reference

```
#include <stl_iterator_base_types.h>
```

Public Types

- using `difference_type` = `ptrdiff_t`
- using `iterator_category` = [random_access_iterator_tag](#)
- using `iterator_concept` = [contiguous_iterator_tag](#)
- using `pointer` = `_Tp *`
- using `reference` = `_Tp &`
- using `value_type` = [remove_cv_t](#)< `_Tp` >

6.653.1 Detailed Description

```
template<typename _Tp>
requires is_object_v<_Tp>
struct std::iterator_traits<_Tp * >
```

Partial specialization for object pointer types.

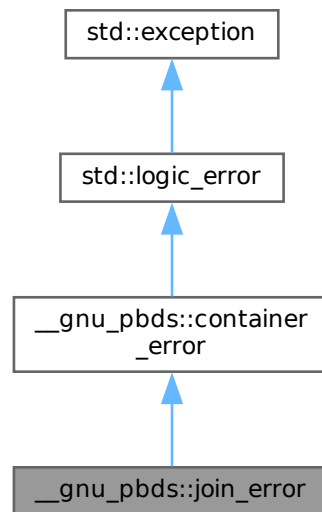
The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.654 __gnu_pbds::join_error Struct Reference

```
#include <exception.hpp>
```

Inheritance diagram for __gnu_pbds::join_error:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.654.1 Detailed Description

A join cannot be performed logical reasons (i.e., the ranges of the two container objects being joined overlaps).

6.654.2 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

6.655 std::jthread Class Reference

```
#include <thread>
```

Public Types

- using **id** = [thread::id](#)
- using **native_handle_type** = thread::native_handle_type

Public Member Functions

- template<typename _Callable, typename... _Args, typename = enable_if_t<!is_same_v<remove_cvref_t<_Callable>, jthread>>>>
 jthread (_Callable &&__f, _Args &&... __args)
- **jthread** (const [jthread](#) &)=delete
- **jthread** ([jthread](#) &&) noexcept=default
- void **detach** ()
- **id** **get_id** () const noexcept
- [stop_source](#) **get_stop_source** () noexcept
- [stop_token](#) **get_stop_token** () const noexcept
- void **join** ()
- bool **joinable** () const noexcept
- native_handle_type **native_handle** ()
- [jthread](#) & **operator=** (const [jthread](#) &)=delete
- [jthread](#) & **operator=** ([jthread](#) &&__other) noexcept
- bool **request_stop** () noexcept
- void **swap** ([jthread](#) &__other) noexcept

Static Public Member Functions

- static unsigned **hardware_concurrency** () noexcept

Friends

- void **swap** ([jthread](#) &__lhs, [jthread](#) &__rhs) noexcept

6.655.1 Detailed Description

A thread with cancellation and automatic joining.

Unlike `std::thread`, destroying a joinable `std::jthread` will not terminate the process. Instead, it will try to request its thread to stop, then will join it.

A `std::jthread` has a `std::stop_source` member which will be passed as the first argument to the callable that runs in the new thread (as long as the callable will accept that argument). That can then be used to send a stop request that the new thread can test for.

Since

C++20

The documentation for this class was generated from the following file:

- [thread](#)

6.656 `__gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >` Class Template Reference

```
#include <left_child_next_sibling_heap_.hpp>
```

Inherits `Cmp_Fn`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `left_child_next_sibling_heap_const_iterator_< node, _Alloc >` **const_iterator**
- typedef `__rebind_v::const_pointer` **const_pointer**
- typedef `__rebind_v::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `const_iterator` **iterator**
- typedef `left_child_next_sibling_heap_node_< Value_Type, Node_Metadata, _Alloc >` **node**
- typedef `left_child_next_sibling_heap_node_point_const_iterator_< node, _Alloc >` **point_const_iterator**
- typedef `point_const_iterator` **point_iterator**
- typedef `__rebind_v::pointer` **pointer**
- typedef `__rebind_v::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- `left_child_next_sibling_heap` (const `Cmp_Fn` &)
- `left_child_next_sibling_heap` (const `left_child_next_sibling_heap` &)
- `iterator` **begin** ()
- `const_iterator` **begin** () const
- void **clear** ()
- bool **empty** () const
- `iterator` **end** ()
- `const_iterator` **end** () const
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- `size_type` **max_size** () const
- `size_type` **size** () const
- void **swap** (`left_child_next_sibling_heap`< `Value_Type`, `Cmp_Fn`, `Node_Metadata`, `_Alloc` > &)

Protected Types

- typedef `alloc_traits::allocator_type` **node_allocator**
- typedef `alloc_traits::const_pointer` **node_const_pointer**
- typedef `Node_Metadata` **node_metadata**
- typedef `alloc_traits::pointer` **node_pointer**
- typedef `std::pair< node_pointer, node_pointer >` **node_pointer_pair**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- node_pointer **get_new_node_for_insert** (const_reference)
- template<typename Pred >
node_pointer **prune** (Pred)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** ([left_child_next_sibling_heap](#) &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_root**
- size_type **m_size**

6.656.1 Detailed Description

template<typename Value_Type, typename Cmp_Fn, typename Node_Metadata, typename _Alloc>
class **__gnu_pbds::detail::left_child_next_sibling_heap**< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >

Base class for a basic heap.

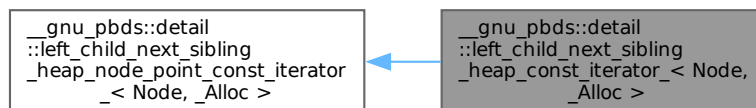
The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_.hpp](#)

6.657 **__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >** Class Template Reference

#include <const_iterator.hpp>

Inheritance diagram for **__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >**:



Public Types

- typedef [base_type::const_pointer](#) **const_pointer**
- typedef [base_type::const_reference](#) **const_reference**
- typedef [_Alloc::difference_type](#) **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**

- typedef `base_type::pointer` `pointer`
- typedef `base_type::reference` `reference`
- typedef `base_type::value_type` `value_type`

Public Member Functions

- `left_child_next_sibling_heap_const_iterator_()`
- `left_child_next_sibling_heap_const_iterator_ (const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > &other)`
- `left_child_next_sibling_heap_const_iterator_ (node_pointer p_nd)`
- `bool operator!= (const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > &other) const`
- `bool operator!= (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`
- `const_reference operator* () const`
- `left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & operator++ ()`
- `left_child_next_sibling_heap_const_iterator_< Node, _Alloc > operator++ (int)`
- `const_pointer operator-> () const`
- `bool operator== (const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > &other) const`
- `bool operator== (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`

Public Attributes

- node_pointer `m_p_nd`

6.657.1 Detailed Description

```
template<typename Node, typename _Alloc>
class __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >
```

Const point-type iterator.

6.657.2 Member Typedef Documentation

`const_pointer`

```
template<typename Node , typename _Alloc >
typedef base_type::const_pointer __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::const_pointer
Iterator's const pointer type.
```

`const_reference`

```
template<typename Node , typename _Alloc >
typedef base_type::const_reference __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::const_reference
Iterator's const reference type.
```

`difference_type`

```
template<typename Node , typename _Alloc >
typedef _Alloc::difference_type __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::difference_type
Difference type.
```

iterator_category

```
template<typename Node , typename _Alloc >
typedef std::forward_iterator_tag __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::iterator_category
Category.
```

pointer

```
template<typename Node , typename _Alloc >
typedef base_type::pointer __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::pointer
Iterator's pointer type.
```

reference

```
template<typename Node , typename _Alloc >
typedef base_type::reference __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Node , typename _Alloc >
typedef base_type::value_type __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_<
Node, _Alloc >::value_type
Iterator's value type.
```

6.657.3 Constructor & Destructor Documentation**left_child_next_sibling_heap_const_iterator_()** [1/2]

```
template<typename Node , typename _Alloc >
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::left_child_↵
next_sibling_heap_const_iterator_ ( ) [inline]
Default constructor.
```

left_child_next_sibling_heap_const_iterator_() [2/2]

```
template<typename Node , typename _Alloc >
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::left_child_↵
next_sibling_heap_const_iterator_ (
    const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & other ) [inline]
Copy constructor.
```

6.657.4 Member Function Documentation**operator"!="()** [1/2]

```
template<typename Node , typename _Alloc >
bool __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::operator!=
(
    const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & other ) const
[inline]
Compares content (negatively) to a different iterator object.
```

operator!=() [2/2]

```
template<typename Node , typename _Alloc >
bool __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::operator!= (
    const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &
    other ) const [inline], [inherited]
```

Compares content (negatively) to a different iterator object.

operator*()

```
template<typename Node , typename _Alloc >
const_reference __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_<
Node, _Alloc >::operator* ( ) const [inline], [inherited]
```

Access.

operator->()

```
template<typename Node , typename _Alloc >
const_pointer __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node,
_Alloc >::operator-> ( ) const [inline], [inherited]
```

Access.

operator==() [1/2]

```
template<typename Node , typename _Alloc >
bool __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::operator==
(
    const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & other ) const
[inline]
```

Compares content to a different iterator object.

operator==() [2/2]

```
template<typename Node , typename _Alloc >
bool __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::operator== (
    const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &
    other ) const [inline], [inherited]
```

Compares content to a different iterator object.

The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_/const_iterator.hpp](#)

6.658 `__gnu_pbds::detail::left_child_next_sibling_heap_node_<_Value, _Metadata, _Alloc>` Struct Template Reference

```
#include <node.hpp>
```

Public Types

- typedef `_Metadata` **metadata_type**
- typedef `rebind_traits<_Alloc, this_type>::pointer` **node_pointer**
- typedef `_Alloc::size_type` **size_type**
- typedef `_Value` **value_type**

Public Attributes

- metadata_type **m_metadata**
- node_pointer **m_p_l_child**
- node_pointer **m_p_next_sibling**
- node_pointer **m_p_prev_or_parent**
- value_type **m_value**

6.658.1 Detailed Description

```
template<typename _Value, typename _Metadata, typename _Alloc>
struct __gnu_pbds::detail::left_child_next_sibling_heap_node_< _Value, _Metadata, _Alloc >
```

Node.

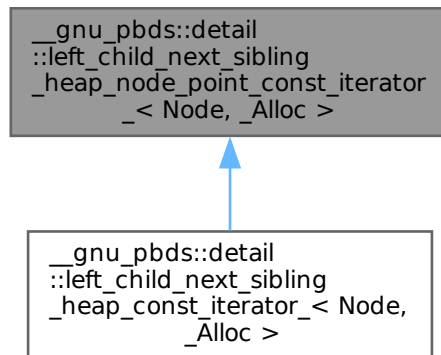
The documentation for this struct was generated from the following file:

- [left_child_next_sibling_heap_/node.hpp](#)

6.659 __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > Class Template Reference

```
#include <point_const_iterator.hpp>
```

Inheritance diagram for __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >:



Public Types

- typedef [rebind_traits](#)< _Alloc, [value_type](#) >::const_pointer [const_pointer](#)
- typedef [rebind_traits](#)< _Alloc, [value_type](#) >::const_reference [const_reference](#)
- typedef [trivial_iterator_difference_type](#) [difference_type](#)
- typedef [trivial_iterator_tag](#) [iterator_category](#)
- typedef [rebind_traits](#)< _Alloc, [value_type](#) >::pointer [pointer](#)
- typedef [rebind_traits](#)< _Alloc, [value_type](#) >::reference [reference](#)
- typedef Node::value_type [value_type](#)

Public Member Functions

- `left_child_next_sibling_heap_node_point_const_iterator_()`
- `left_child_next_sibling_heap_node_point_const_iterator_(const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other)`
- `left_child_next_sibling_heap_node_point_const_iterator_(node_pointer p_nd)`
- `bool operator!= (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`
- `const_reference operator* () const`
- `const_pointer operator-> () const`
- `bool operator== (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`

Public Attributes

- node_pointer `m_p_nd`

Protected Types

- typedef `rebind_traits< _Alloc, Node >::pointer` `node_pointer`

6.659.1 Detailed Description

`template<typename Node, typename _Alloc>`

`class __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >`

Const point-type iterator.

6.659.2 Member Typedef Documentation

`const_pointer`

`template<typename Node , typename _Alloc >`

typedef `rebind_traits<_Alloc,value_type>::const_pointer` `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::const_pointer`

Iterator's const pointer type.

`const_reference`

`template<typename Node , typename _Alloc >`

typedef `rebind_traits<_Alloc,value_type>::const_reference` `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::const_reference`

Iterator's const reference type.

`difference_type`

`template<typename Node , typename _Alloc >`

typedef `trivial_iterator_difference_type` `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::difference_type`

Difference type.

`iterator_category`

`template<typename Node , typename _Alloc >`

typedef `trivial_iterator_tag` `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::iterator_category`

Category.

pointer

```
template<typename Node , typename _Alloc >
typedef rebind\_traits<_Alloc,value_type>::pointer \_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_co
Node, _Alloc >::pointer
Iterator's pointer type.
```

reference

```
template<typename Node , typename _Alloc >
typedef rebind\_traits<_Alloc,value_type>::reference \_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_co
Node, _Alloc >::reference
Iterator's reference type.
```

value_type

```
template<typename Node , typename _Alloc >
typedef Node::value_type \_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_<
Node, _Alloc >::value_type
Iterator's value type.
```

6.659.3 Constructor & Destructor Documentation**[left_child_next_sibling_heap_node_point_const_iterator_\(\)](#) [1/2]**

```
template<typename Node , typename _Alloc >
\_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_< Node, _Alloc >↵
::left_child_next_sibling_heap_node_point_const_iterator_ ( ) [inline]
Default constructor.
```

[left_child_next_sibling_heap_node_point_const_iterator_\(\)](#) [2/2]

```
template<typename Node , typename _Alloc >
\_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_< Node, _Alloc >↵
::left_child_next_sibling_heap_node_point_const_iterator_ (
    const left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_< Node, _Alloc > &
    other ) [inline]
Copy constructor.
```

6.659.4 Member Function Documentation**[operator"!="\(\)](#)**

```
template<typename Node , typename _Alloc >
bool \_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_< Node, _Alloc
>::operator!= (
    const left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_< Node, _Alloc > &
    other ) const [inline]
Compares content (negatively) to a different iterator object.
```

[operator*\(\)](#)

```
template<typename Node , typename _Alloc >
const\_reference \_\_gnu\_pbds::detail::left\_child\_next\_sibling\_heap\_node\_point\_const\_iterator\_<
Node, _Alloc >::operator* ( ) const [inline]
Access.
```

operator->()

```
template<typename Node , typename _Alloc >
const_pointer __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node,
_Alloc >::operator-> ( ) const [inline]
Access.
```

operator==()

```
template<typename Node , typename _Alloc >
bool __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::operator== (
    const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &
    other ) const [inline]
```

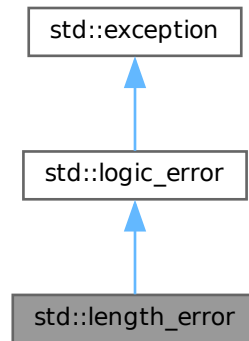
Compares content to a different iterator object.

The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_/point_const_iterator.hpp](#)

6.660 std::length_error Class Reference

Inheritance diagram for std::length_error:

**Public Member Functions**

- **length_error** (const char *) _GLIBCXX_TXN_SAFE
- **length_error** (const [length_error](#) &)=default
- **length_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **length_error** ([length_error](#) &&)=default
- [length_error](#) & **operator=** (const [length_error](#) &)=default
- [length_error](#) & **operator=** ([length_error](#) &&)=default
- virtual const char * **what** () const noexcept

6.660.1 Detailed Description

Thrown when an object is constructed that would exceed its maximum permitted size (e.g., a basic_string instance).

6.660.2 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

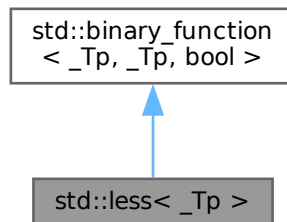
The documentation for this class was generated from the following file:

- [stdexcept](#)

6.661 std::less<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::less<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `bool` **operator()** (const `_Tp` &__x, const `_Tp` &__y) const

6.661.1 Detailed Description

```
template<typename _Tp>
```

```
struct std::less<_Tp>
```

One of the [comparison functors](#).

6.661.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type [inherited]
```

`first_argument_type` is the type of the first argument

result_type

typedef bool [std::binary_function](#)< _Tp , _Tp , bool >::result_type [inherited]
 result_type is the return type

second_argument_type

typedef _Tp [std::binary_function](#)< _Tp , _Tp , bool >::second_argument_type [inherited]
 second_argument_type is the type of the second argument
 The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.662 std::ranges::less Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using **is_transparent** = __is_transparent

Public Member Functions

- template<typename _Tp , typename _Up >
 requires totally_ordered_with<_Tp, _Up>
 constexpr bool **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept([std::declval](#)< _Tp >())<
[std::declval](#)< _Up >()))

6.662.1 Detailed Description

ranges::less function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.663 std::less< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp , typename _Up >
 constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept([std::forward](#)< _Tp >(__t)<
[std::forward](#)< _Up >(__u))) -> decltype([std::forward](#)< _Tp >(__t)< [std::forward](#)< _Up >(__u))
- template<typename _Tp , typename _Up >
 constexpr bool **operator()** (_Tp *__t, _Up *__u) const noexcept

6.663.1 Detailed Description

One of the [comparison functors](#).

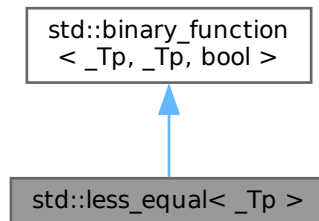
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.664 std::less_equal< _Tp > Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::less_equal< _Tp >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- constexpr bool **operator()** (const _Tp &__x, const _Tp &__y) const

6.664.1 Detailed Description

```
template<typename _Tp>
struct std::less_equal< _Tp >
```

One of the [comparison functors](#).

6.664.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary\_function< _Tp , _Tp , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
The documentation for this struct was generated from the following file:
```

- [stl_function.h](#)

6.665 std::ranges::less_equal Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using **is_transparent** = __is_transparent

Public Member Functions

- template<typename _Tp, typename _Up >
requires totally_ordered_with<_Tp, _Up>
constexpr bool **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::declval<_Up >())<std::declval<_Tp >()))

6.665.1 Detailed Description

ranges::less_equal function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.666 std::less_equal< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp >(__t)<=std::forward<_Up >(__u))) -> decltype(std::forward<_Tp >(__t)<=std::forward<_Up >(__u))
- template<typename _Tp, typename _Up >
constexpr bool **operator()** (_Tp *__t, _Up *__u) const noexcept

6.666.1 Detailed Description

One of the [comparison functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.667 __gnu_cxx::limit_condition::limit_adjutor Struct Reference

```
#include <throw_allocator.h>
```

Inherits [__gnu_cxx::limit_condition::adjutor_base](#).

Public Member Functions

- **limit_adjutor** (const size_t __l)

6.667.1 Detailed Description

Enter the nth condition.

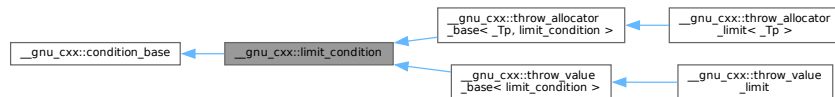
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.668 __gnu_cxx::limit_condition Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for __gnu_cxx::limit_condition:



Classes

- struct [always_adjustor](#)
- struct [limit_adjustor](#)
- struct [never_adjustor](#)

Static Public Member Functions

- static `size_t` & **count** ()
- static `size_t` & **limit** ()
- static void **set_limit** (const `size_t` __l)
- static void **throw_conditionally** ()

6.668.1 Detailed Description

Base class for incremental control and throw.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.669 std::linear_congruential_engine<_UIntType, __a, __c, __m> Class Template Reference

```
#include <random.h>
```

Public Types

- typedef `_UIntType` [result_type](#)

Public Member Functions

- [linear_congruential_engine](#) ()
- template<typename `_Sseq`, typename `=_If_seed_seq<_Sseq>>`
[linear_congruential_engine](#) (`_Sseq` &__q)
- [linear_congruential_engine](#) ([result_type](#) __s)

- void `discard` (unsigned long long __z)
- `result_type` `operator()` ()
- template<typename _Sseq >
_If_seed_seq< _Sseq > `seed` (_Sseq &__q)
- template<typename _Sseq >
auto `seed` (_Sseq &__q) -> _If_seed_seq< _Sseq >
- void `seed` (`result_type` __s=default_seed)

Static Public Member Functions

- static constexpr `result_type` `max` ()
- static constexpr `result_type` `min` ()

Static Public Attributes

- static constexpr `result_type` `default_seed`
- static constexpr `result_type` `increment`
- static constexpr `result_type` `modulus`
- static constexpr `result_type` `multiplier`

Friends

- template<typename _UIntType1 , _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT , typename _Traits >
`std::basic_ostream`< _CharT, _Traits > & `operator<<` (`std::basic_ostream`< _CharT, _Traits > &__os, const `std::linear_congruential_engine`< _UIntType1, __a1, __c1, __m1 > &__lcr)
- bool `operator==` (const `linear_congruential_engine` &__lhs, const `linear_congruential_engine` &__rhs)
- template<typename _UIntType1 , _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT , typename _Traits >
`std::basic_istream`< _CharT, _Traits > & `operator>>` (`std::basic_istream`< _CharT, _Traits > &__is, `std::linear_congruential_engine`< _UIntType1, __a1, __c1, __m1 > &__lcr)

6.669.1 Detailed Description

template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>
class `std::linear_congruential_engine`< _UIntType, __a, __c, __m >

A model of a linear congruential random number generator.

A random number generator that produces pseudorandom numbers via linear function:

$$x_{i+1} \leftarrow (ax_i + c) \bmod m$$

The template parameter `_UIntType` must be an unsigned integral type large enough to store values up to (`__m`-1). If the template parameter `__m` is 0, the modulus `__m` used is `std::numeric_limits<_UIntType>::max()` plus 1. Otherwise, the template parameters `__a` and `__c` must be less than `__m`.

The size of the state is 1.

6.669.2 Member Typedef Documentation

`result_type`

template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
typedef _UIntType `std::linear_congruential_engine`< _UIntType, __a, __c, __m >::`result_type`

The type of the generated random value.

6.669.3 Constructor & Destructor Documentation

linear_congruential_engine() [1/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine ( ) [inline]
Constructs a linear_congruential_engine random number generator engine with seed 1.
```

linear_congruential_engine() [2/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine (
    result_type __s ) [inline], [explicit]
Constructs a linear_congruential_engine random number generator engine with seed __s. The default seed value is 1.
```

Parameters

<code>__s</code>	The initial seed value.
------------------	-------------------------

References [std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed\(\)](#).

linear_congruential_engine() [3/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine (
    _Sseq & __q ) [inline], [explicit]
Constructs a linear_congruential_engine random number generator engine seeded from the seed sequence __q.
```

Parameters

<code>__q</code>	the seed sequence.
------------------	--------------------

References [std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed\(\)](#).

6.669.4 Member Function Documentation

discard()

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
void std::linear_congruential_engine< _UIntType, __a, __c, __m >::discard (
    unsigned long long __z ) [inline]
Discard a sequence of random numbers.
```

max()

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
static constexpr result_type std::linear_congruential_engine< _UIntType, __a, __c, __m >::max ( )
[inline], [static], [constexpr]
Gets the largest possible value in the output range.
```

min()

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
```

```
static constexpr result_type std::linear_congruential_engine< _UIntType, __a, __c, __m >::min ( )
[inline], [static], [constexpr]
```

Gets the smallest possible value in the output range.

The minimum depends on the __c parameter: if it is zero, the minimum generated must be > 0, otherwise 0 is allowed.

operator>()()

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
result_type std::linear_congruential_engine< _UIntType, __a, __c, __m >::operator() ( ) [inline]
```

Gets the next random number in the sequence.

seed() [1/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
template<typename _Sseq >
_If_seed_seq< _Sseq > std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed (
    _Sseq & __q )
```

Reseeds the linear_congruential_engine random number generator engine sequence using values from the seed sequence __q.

Parameters

<code>__q</code>	the seed sequence.
------------------	--------------------

seed() [2/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
template<typename _Sseq >
auto std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed (
    _Sseq & __q ) -> _If_seed_seq<_Sseq>
```

Seeds the LCR engine with a value generated by __q.

References [std::__lg\(\)](#).

seed() [3/3]

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
void std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed (
    result_type __s = default_seed )
```

Reseeds the linear_congruential_engine random number generator engine sequence to the seed __s.

Parameters

<code>__s</code>	The new seed.
------------------	---------------

Seeds the LCR with integral value __s, adjusted so that the ring identity is never a member of the convergence set.

Referenced by [std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine\(\)](#), and [std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine\(\)](#).

6.669.5 Friends And Related Symbol Documentation

operator<<

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
template<typename _UIntType1 , _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT
, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::linear_congruential_engine< _UIntType1, __a1, __c1, __m1 > & __lcr )
[friend]
```

Writes the textual representation of the state x(i) of x to __os.

Parameters

<code>__os</code>	The output stream.
<code>__lcr</code>	A % linear_congruential_engine random number generator.

Returns

`__os`.

operator==

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
bool operator== (
    const std::linear_congruential_engine< _UIntType, __a, __c, __m > & __lhs,
    const std::linear_congruential_engine< _UIntType, __a, __c, __m > & __rhs ) [friend]
```

Compares two linear congruential random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A linear congruential random number generator object.
<code>__rhs</code>	Another linear congruential random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

operator>>

```
template<typename _UIntType , _UIntType __a, _UIntType __c, _UIntType __m>
template<typename _UIntType1 , _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT
, typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::linear_congruential_engine< _UIntType1, __a1, __c1, __m1 > & __lcr ) [friend]
```

Sets the state of the engine by reading its textual representation from __is.

The textual representation must have been previously written using an output stream whose imbued locale and whose type's template specialization arguments `_CharT` and `_Traits` were the same as those of `__is`.

Parameters

<code>__is</code>	The input stream.
<code>__lcr</code>	A % linear_congruential_engine random number generator.

Returns

`__is`.

6.669.6 Member Data Documentation

increment

```
template<typename UIntType , UIntType __a, UIntType __c, UIntType __m>
constexpr result_type std::linear_congruential_engine< UIntType, __a, __c, __m >::increment
[static], [constexpr]
```

An increment.

modulus

```
template<typename UIntType , UIntType __a, UIntType __c, UIntType __m>
constexpr result_type std::linear_congruential_engine< UIntType, __a, __c, __m >::modulus [static],
[constexpr]
```

The modulus.

multiplier

```
template<typename UIntType , UIntType __a, UIntType __c, UIntType __m>
constexpr result_type std::linear_congruential_engine< UIntType, __a, __c, __m >::multiplier
[static], [constexpr]
```

The multiplier.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.670 `__gnu_pbds::linear_probe_fn< Size_Type >` Class Template Reference

```
#include <hash_policy.hpp>
```

Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- void **swap** ([linear_probe_fn< Size_Type >](#) &other)

Protected Member Functions

- `size_type` **operator()** (`size_type` i) const

6.670.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::linear_probe_fn< Size_Type >
```

A probe sequence policy using fixed increments.

6.670.2 Member Function Documentation

operator()()

```
template<typename Size_Type = std::size_t>
size_type __gnu_pbds::linear_probe_fn< Size_Type >::operator() (
    size_type i ) const [inline], [protected]
```

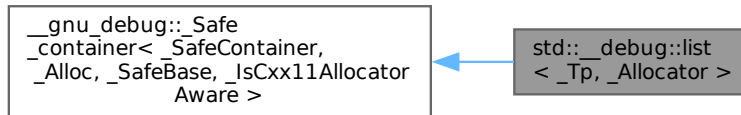
Returns the i-th offset from the hash value.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.671 std::__debug::list< _Tp, _Allocator > Class Template Reference

Inheritance diagram for std::__debug::list< _Tp, _Allocator >:



Public Types

- typedef _Allocator **allocator_type**
- typedef __gnu_debug::Safe_iterator< _Base_const_iterator, list > **const_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef std::reverse_iterator< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef __gnu_debug::Safe_iterator< _Base_iterator, list > **iterator**
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef std::reverse_iterator< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **list** (_Base_ref __x)
- template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>>
list (_InputIterator __first, _InputIterator __last, const _Allocator &__a= _Allocator())
- **list** (const _Allocator &__a) noexcept

- **list** (const [list](#) &)=default
- **list** (const [list](#) &__x, const __type_identity_t< allocator_type > &__a)
- **list** ([initializer_list](#)< value_type > __l, const allocator_type &__a=allocator_type())
- **list** ([list](#) &&)=default
- **list** ([list](#) &&__x, const __type_identity_t< allocator_type > &__a) noexcept([std::is_nothrow_constructible](#)< [_Base](#), [_Base](#), const allocator_type & >::value)
- **list** (size_type __n, const __type_identity_t< _Tp > &__value, const _Allocator &__a=_Allocator())
- **list** (size_type __n, const allocator_type &__a=allocator_type())
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** ([initializer_list](#)< value_type > __l)
- void **assign** (size_type __n, const _Tp &__t)
- const_reference **back** () const noexcept
- reference **back** () noexcept
- const_iterator **begin** () const noexcept
- iterator **begin** () noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
[iterator](#) **emplace** (const_iterator __position, _Args &&... __args)
- const_iterator **end** () const noexcept
- iterator **end** () noexcept
- iterator **erase** (const_iterator __first, const_iterator __last) noexcept
- iterator **erase** (const_iterator __position) noexcept
- const_reference **front** () const noexcept
- reference **front** () noexcept
- iterator **insert** (const_iterator __p, [initializer_list](#)< value_type > __l)
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
[iterator](#) **insert** (const_iterator __position, _InputIterator __first, _InputIterator __last)
- iterator **insert** (const_iterator __position, _Tp &&__x)
- iterator **insert** (const_iterator __position, const _Tp &__x)
- iterator **insert** (const_iterator __position, size_type __n, const _Tp &__x)
- void **merge** ([list](#) &&__x)
- template<class _Compare >
void **merge** ([list](#) &&__x, _Compare __comp)
- void **merge** ([list](#) &__x)
- template<typename _Compare >
void **merge** ([list](#) &__x, _Compare __comp)
- [list](#) & **operator=** (const [list](#) &)=default
- [list](#) & **operator=** ([initializer_list](#)< value_type > __l)
- [list](#) & **operator=** ([list](#) &&)=default
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- const_reverse_iterator **rbegin** () const noexcept
- reverse_iterator **rbegin** () noexcept
- __remove_return_type **remove** (const _Tp &__value)

- `template<class _Predicate >`
`__remove_return_type remove_if (_Predicate __pred)`
- `const_reverse_iterator rend ()` `const noexcept`
- `reverse_iterator rend ()` `noexcept`
- `void resize (size_type __sz)`
- `void resize (size_type __sz, const _Tp &__c)`
- `void sort ()`
- `template<typename _StrictWeakOrdering >`
`void sort (_StrictWeakOrdering __pred)`
- `void splice (const_iterator __position, list &&__x)` `noexcept`
- `void splice (const_iterator __position, list &&__x, const_iterator __first, const_iterator __last)` `noexcept`
- `void splice (const_iterator __position, list &&__x, const_iterator __i)` `noexcept`
- `void splice (const_iterator __position, list &__x)` `noexcept`
- `void splice (const_iterator __position, list &__x, const_iterator __first, const_iterator __last)` `noexcept`
- `void splice (const_iterator __position, list &__x, const_iterator __i)` `noexcept`
- `void swap (list &__x)` `noexcept` (`/*conditional */`)
- `__remove_return_type unique ()`
- `template<class _BinaryPredicate >`
`__remove_return_type unique (_BinaryPredicate __binary_pred)`

Protected Member Functions

- `void _M_swap (_Safe_container &__x)` `noexcept`

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >`
`class ::__gnu_debug::_Safe_iterator`

6.671.1 Detailed Description

`template<typename _Tp, typename _Allocator = std::allocator<_Tp>>`

`class std::__debug::list< _Tp, _Allocator >`

Class `std::list` with safety/checking/debug instrumentation.

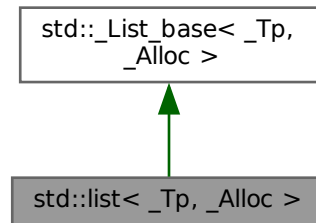
The documentation for this class was generated from the following file:

- [debug/list](#)

6.672 std::list< _Tp, _Alloc > Class Template Reference

```
#include <stl_list.h>
```

Inheritance diagram for std::list< _Tp, _Alloc >:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_List_const_iterator`< `_Tp` > **const_iterator**
- typedef `_Tp_alloc_traits::const_pointer` **const_pointer**
- typedef `_Tp_alloc_traits::const_reference` **const_reference**
- typedef `std::reverse_iterator`< `const_iterator` > **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `_List_iterator`< `_Tp` > **iterator**
- typedef `_Tp_alloc_traits::pointer` **pointer**
- typedef `_Tp_alloc_traits::reference` **reference**
- typedef `std::reverse_iterator`< `iterator` > **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `list` ()=default
- template<typename `_InputIterator` , typename = `std::_RequireInputIter`< `_InputIterator` >>
`list` (`_InputIterator` __first, `_InputIterator` __last, const `allocator_type` &__a=`allocator_type`())
- `list` (const `allocator_type` &__a) noexcept
- `list` (const `list` &__x)
- `list` (const `list` &__x, const `__type_identity_t`< `allocator_type` > &__a)
- `list` (`initializer_list`< `value_type` > __l, const `allocator_type` &__a=`allocator_type`())
- `list` (`list` &&)=default
- `list` (`list` &&__x, const `__type_identity_t`< `allocator_type` > &__a) noexcept(`_Node_alloc_traits::S_always_`↵
`equal`())
- `list` (`size_type` __n, const `allocator_type` &__a=`allocator_type`())
- `list` (`size_type` __n, const `value_type` &__value, const `allocator_type` &__a=`allocator_type`())
- `~list` ()=default
- template<typename `_InputIterator` , typename = `std::_RequireInputIter`< `_InputIterator` >>
void `assign` (`_InputIterator` __first, `_InputIterator` __last)
- void `assign` (`initializer_list`< `value_type` > __l)
- void `assign` (`size_type` __n, const `value_type` &__val)
- const_reference `back` () const noexcept

- reference [back](#) () noexcept
- [const_iterator begin](#) () const noexcept
- [iterator begin](#) () noexcept
- [const_iterator cbegin](#) () const noexcept
- [const_iterator cend](#) () const noexcept
- void [clear](#) () noexcept
- [const_reverse_iterator crbegin](#) () const noexcept
- [const_reverse_iterator crend](#) () const noexcept
- template<typename... _Args>
 [iterator emplace](#) ([const_iterator](#) __position, _Args &&... __args)
- template<typename... _Args>
 reference [emplace_back](#) (_Args &&... __args)
- template<typename... _Args>
 reference [emplace_front](#) (_Args &&... __args)
- bool [empty](#) () const noexcept
- [const_iterator end](#) () const noexcept
- [iterator end](#) () noexcept
- [iterator erase](#) ([const_iterator](#) __first, [const_iterator](#) __last) noexcept
- [iterator erase](#) ([const_iterator](#) __position) noexcept
- const_reference [front](#) () const noexcept
- reference [front](#) () noexcept
- allocator_type [get_allocator](#) () const noexcept
- [iterator insert](#) ([const_iterator](#) __p, [initializer_list](#)< value_type > __l)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>
 [iterator insert](#) ([const_iterator](#) __position, _InputIterator __first, _InputIterator __last)
- [iterator insert](#) ([const_iterator](#) __position, const value_type &__x)
- [iterator insert](#) ([const_iterator](#) __position, size_type __n, const value_type &__x)
- [iterator insert](#) ([const_iterator](#) __position, value_type &&__x)
- size_type [max_size](#) () const noexcept
- void [merge](#) ([list](#) &&__x)
- template<typename _StrictWeakOrdering >
 void [merge](#) ([list](#) &&__x, _StrictWeakOrdering __comp)
- void [merge](#) ([list](#) &&__x)
- template<typename _StrictWeakOrdering >
 void [merge](#) ([list](#) &__x, _StrictWeakOrdering __comp)
- [list](#) & [operator=](#) (const [list](#) &__x)
- [list](#) & [operator=](#) ([initializer_list](#)< value_type > __l)
- [list](#) & [operator=](#) ([list](#) &&__x) noexcept(_Node_alloc_traits::_S_nothrow_move())
- void [pop_back](#) () noexcept
- void [pop_front](#) () noexcept
- void [push_back](#) (const value_type &__x)
- void [push_back](#) (value_type &&__x)
- void [push_front](#) (const value_type &__x)
- void [push_front](#) (value_type &&__x)
- [const_reverse_iterator rbegin](#) () const noexcept
- [reverse_iterator rbegin](#) () noexcept
- __remove_return_type [remove](#) (const _Tp &__value)
- template<typename _Predicate >
 __remove_return_type [remove_if](#) (_Predicate)
- [const_reverse_iterator rend](#) () const noexcept
- [reverse_iterator rend](#) () noexcept

- void [resize](#) (size_type __new_size)
- void [resize](#) (size_type __new_size, const value_type &__x)
- void [reverse](#) () noexcept
- size_type [size](#) () const noexcept
- void [sort](#) ()
- template<typename _StrictWeakOrdering >
void [sort](#) (_StrictWeakOrdering)
- void [splice](#) (const_iterator __position, list &&__x) noexcept
- void [splice](#) (const_iterator __position, list &&__x, const_iterator __first, const_iterator __last) noexcept
- void [splice](#) (const_iterator __position, list &&__x, const_iterator __i) noexcept
- void [splice](#) (const_iterator __position, list &__x) noexcept
- void [splice](#) (const_iterator __position, list &__x, const_iterator __first, const_iterator __last) noexcept
- void [splice](#) (const_iterator __position, list &__x, const_iterator __i) noexcept
- void [swap](#) (list &__x) noexcept
- __remove_return_type [unique](#) ()
- template<typename _BinaryPredicate >
__remove_return_type [unique](#) (_BinaryPredicate)

Protected Types

- typedef [_List_node](#)< _Tp > [_Node](#)

Protected Member Functions

- template<typename _InputIterator >
void [_M_assign_dispatch](#) (_InputIterator __first, _InputIterator __last, __false_type)
- template<typename _Integer >
void [_M_assign_dispatch](#) (_Integer __n, _Integer __val, __true_type)
- void [_M_check_equal_allocators](#) (list &__x) noexcept
- void [_M_clear](#) () noexcept
- template<typename... _Args>
[_Node](#) * [_M_create_node](#) (_Args &&... __args)
- void [_M_dec_size](#) (size_t)
- void [_M_default_append](#) (size_type __n)
- void [_M_default_initialize](#) (size_type __n)
- size_t [_M_distance](#) (const void *, const void *) const
- void [_M_erase](#) (iterator __position) noexcept
- void [_M_fill_assign](#) (size_type __n, const value_type &__val)
- void [_M_fill_initialize](#) (size_type __n, const value_type &__x)
- [_Node_alloc_traits::pointer](#) [_M_get_node](#) ()
- const [_Node_alloc_type](#) & [_M_get_Node_allocator](#) () const noexcept
- [_Node_alloc_type](#) & [_M_get_Node_allocator](#) () noexcept
- size_t [_M_get_size](#) () const
- void [_M_inc_size](#) (size_t)
- void [_M_init](#) () noexcept
- template<typename _InputIterator >
void [_M_initialize_dispatch](#) (_InputIterator __first, _InputIterator __last, __false_type)
- template<typename _Integer >
void [_M_initialize_dispatch](#) (_Integer __n, _Integer __x, __true_type)
- template<typename... _Args>
void [_M_insert](#) (iterator __position, _Args &&... __args)
- void [_M_move_assign](#) (list &&__x, [false_type](#))

- void **_M_move_assign** (list &&__x, true_type) noexcept
- void **_M_move_nodes** (_List_base &&__x)
- size_t **_M_node_count** () const
- void **_M_put_node** (typename _Node_alloc_traits::pointer __p) noexcept
- **const_iterator** **_M_resize_pos** (size_type &__new_size) const
- void **_M_set_size** (size_t)
- void **_M_transfer** (iterator __position, iterator __first, iterator __last)

Static Protected Member Functions

- static size_t **_S_distance** (const __detail::_List_node_base *__first, const __detail::_List_node_base *__last)
- static size_t **_S_distance** (const_iterator, const_iterator)

Protected Attributes

- _List_impl _M_impl

6.672.1 Detailed Description

template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::list<_Tp, _Alloc >

A standard container with linear time access to elements, and fixed time insertion/deletion at any point in the sequence.

Template Parameters

<i>_Tp</i>	Type of element.
<i>_Alloc</i>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#) with the exception of `at` and `operator[]`.

This is a *doubly linked* list. Traversal up and down the list requires linear time, but adding and removing elements (or *nodes*) is done in constant time, regardless of where the change takes place. Unlike `std::vector` and `std::deque`, random-access iterators are not provided, so subscripting (`[]`) access is not allowed. For algorithms which only need sequential access, this lack makes no difference.

Also unlike the other standard containers, `std::list` provides specialized algorithms unique to linked lists, such as splicing, sorting, and in-place reversal.

A couple points on memory allocation for `list<Tp>`:

First, we never actually allocate a `Tp`, we allocate `List_node<Tp>`'s and trust [20.1.5]/4 to DTRT. This is to ensure that after elements from `list<X,Alloc1>` are spliced into `list<X,Alloc2>`, destroying the memory of the second list is a valid operation, i.e., `Alloc1` giveth and `Alloc2` taketh away.

Second, a list conceptually represented as

A <---> B <---> C <---> D

is actually circular; a link exists between A and D. The list class holds (as its only data member) a private `list::iterator` pointing to *D*, not to *A*! To get to the head of the list, we start at the tail and move forward by one. When this member iterator's `next/previous` pointers refer to itself, the list is empty.

6.672.2 Constructor & Destructor Documentation

list() [1/8]


```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc >::list ( ) [default]
Creates a list with no elements.
```

list() [2/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc >::list (
    const allocator_type & __a ) [inline], [explicit], [noexcept]
```

Creates a list with no elements.

Parameters



 <code>__a</code>	An allocator object.
--	----------------------

list() [3/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc >::list (
    size_type __n,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Creates a list with default constructed elements.

Parameters

 <code>__n</code>	The number of elements to initially create.
 <code>__a</code>	An allocator object.

This constructor fills the list with `__n` default constructed elements.

list() [4/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc >::list (
    size_type __n,
    const value_type & __value,
    const allocator_type & __a = allocator_type() ) [inline]
```

Creates a list with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__a</code>	An allocator object.

This constructor fills the list with `__n` copies of `__value`.

list() [5/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc >::list (
    const list<_Tp, _Alloc > & __x ) [inline]
```

List copy constructor.

Parameters

<code>__x</code>	A list of identical element and allocator types.
------------------	--

The newly-created list uses a copy of the allocation object used by `__x` (unless the allocator traits dictate a different object).

References `std::list< _Tp, _Alloc >::begin()`, and `std::list< _Tp, _Alloc >::end()`.

list() [6/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list< _Tp, _Alloc >::list (
    list< _Tp, _Alloc > && ) [default]
```

List move constructor.

The newly-created list contains the exact contents of the moved instance. The contents of the moved instance are a valid, but unspecified list.

list() [7/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list< _Tp, _Alloc >::list (
    initializer_list< value_type > __l,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a list from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> of <code>value_type</code> .
<code>__a</code>	An allocator object.

Create a list consisting of copies of the elements in the `initializer_list __l`. This is linear in `__l.size()`.

list() [8/8]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
std::list< _Tp, _Alloc >::list (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a list from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__a</code>	An allocator object.

Create a list consisting of copies of the elements from `[__first, __last)`. This is linear in `N` (where `N` is `distance(__first, __last)`).

~list()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::list<_Tp, _Alloc>::~~list ( ) [default]
```

No explicit dtor needed as the _Base dtor takes care of things. The _Base dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

6.672.3 Member Function Documentation**_M_create_node()**

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename... _Args>
_Node * std::list<_Tp, _Alloc>::_M_create_node (
    _Args &&... __args ) [inline], [protected]
```

Parameters

<code>__args</code>	An instance of user data.
---------------------	---------------------------

Allocates space for a new node and constructs a copy of `__args` in it.

assign() [1/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::RequireInputIter<_InputIterator>>
void std::list<_Tp, _Alloc>::assign (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Assigns a range to a list.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a list with copies of the elements in the range `[__first, __last)`.

Note that the assignment completely changes the list and that the resulting list's size is the same as the number of elements assigned.

assign() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::assign (
    initializer_list<value_type> __l ) [inline]
```

Assigns an initializer_list to a list.

Parameters

<code>↵</code>	An initializer_list of value_type.
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code>/</code>	

Replace the contents of the list with copies of the elements in the initializer_list __l/. This is linear in __l.size().

assign() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::assign (
    size_type __n,
    const value_type & __val ) [inline]
```

Assigns a given value to a list.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a list with __n copies of the given value. Note that the assignment completely changes the list and that the resulting list's size is the same as the number of elements assigned.

Referenced by [std::list< _Tp, _Alloc >::operator=\(\)](#).

back() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reference std::list< _Tp, _Alloc >::back ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the last element of the list.

References [std::list< _Tp, _Alloc >::end\(\)](#).

back() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reference std::list< _Tp, _Alloc >::back ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the last element of the list.

References [std::list< _Tp, _Alloc >::end\(\)](#).

begin() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::list< _Tp, _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the list. Iteration is done in ordinary element order.

begin() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::list< _Tp, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the list. Iteration is done in ordinary element order.

Referenced by [std::list< _Tp, _Alloc >::list\(\)](#), [std::list< _Tp, _Alloc >::crend\(\)](#), [std::list< _Tp, _Alloc >::front\(\)](#), [std::list< _Tp, _Alloc >::front\(\)](#), [std::list< _Tp, _Alloc >::insert\(\)](#), [std::list< _Tp, _Alloc >::insert\(\)](#), [std::list< _Tp, _Alloc >::merge\(\)](#), [std::list< _Tp, _Alloc >::merge\(\)](#), [std::operator<\(\)](#), [std::list< _Tp, _Alloc >::operator=\(\)](#), [std::operator==\(\)](#), [std::list< _Tp, _Alloc >::pop_f](#), [std::list< _Tp, _Alloc >::push_front\(\)](#), [std::list< _Tp, _Alloc >::remove\(\)](#), [std::list< _Tp, _Alloc >::remove_if\(\)](#), [std::list< _Tp, _Alloc >::rend\(\)](#), [std::list< _Tp, _Alloc >::rend\(\)](#), [std::list< _Tp, _Alloc >::splice\(\)](#), [std::list< _Tp, _Alloc >::unique\(\)](#), and [std::list< _Tp, _Alloc >::unique\(\)](#).

cbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::list< _Tp, _Alloc >::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the list. Iteration is done in ordinary element order.

cend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::list< _Tp, _Alloc >::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the list. Iteration is done in ordinary element order.

clear()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

crbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::list< _Tp, _Alloc >::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the list. Iteration is done in reverse element order.

References `std::list<_Tp, _Alloc>::end()`.

crend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::list< _Tp, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

References `std::list<_Tp, _Alloc>::begin()`.

emplace()

```
template<typename _Tp , typename _Alloc >
template<typename... _Args>
list< _Tp, _Alloc >::iterator list::emplace (
    const_iterator __position,
    _Args &&... __args )
```

Constructs object in list before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the list.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type T constructed with `T(std::forward<Args>(args)...) before the specified location. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references. Referenced by std::list<_Tp, _Alloc>::insert().`

empty()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
bool std::list<_Tp, _Alloc>::empty ( ) const [inline], [noexcept]
```

Returns true if the list is empty. (Thus `begin()` would equal `end()`.)
Referenced by `std::list<_Tp, _Alloc>::insert()`, and `std::list<_Tp, _Alloc>::splice()`.

end() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_iterator std::list<_Tp, _Alloc>::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the list. Iteration is done in ordinary element order.

end() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::list<_Tp, _Alloc>::end ( ) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the list. Iteration is done in ordinary element order.
Referenced by `std::list<_Tp, _Alloc>::list()`, `std::list<_Tp, _Alloc>::back()`, `std::list<_Tp, _Alloc>::back()`, `std::list<_Tp, _Alloc>::cbegin()`, `std::list<_Tp, _Alloc>::merge()`, `std::list<_Tp, _Alloc>::merge()`, `std::operator<()`, `std::list<_Tp, _Alloc>::operator=()`, `std::operator==()`, `std::list<_Tp, _Alloc>::push_back()`, `std::list<_Tp, _Alloc>::rbegin()`, `std::list<_Tp, _Alloc>::rbegin()`, and `std::list<_Tp, _Alloc>::splice()`.

erase() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::list<_Tp, _Alloc>::erase (
    const_iterator __first,
    const_iterator __last ) [inline], [noexcept]
```

Remove a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by *last* prior to erasing (or `end()`).

This function will erase the elements in the range `[first,last)` and shorten the list accordingly.

This operation is linear time in the size of the range and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References `std::list<_Tp, _Alloc>::erase()`.

erase() [2/2]

```
template<typename _Tp , typename _Alloc >
list< _Tp, _Alloc >::iterator list::erase (
    const_iterator __position ) [noexcept]
```

Remove element at given position.

Parameters

<code>__position</code>	Iterator pointing to element to be erased.
-------------------------	--

Returns

An iterator pointing to the next element (or end()).

This function will erase the element at the given position and thus shorten the list by one.

Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Referenced by [std::list<_Tp, _Alloc>::erase\(\)](#).

front() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reference std::list< _Tp, _Alloc >::front ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the list.

References [std::list<_Tp, _Alloc>::begin\(\)](#).

front() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reference std::list< _Tp, _Alloc >::front ( ) [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the list.

References [std::list<_Tp, _Alloc>::begin\(\)](#).

get_allocator()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
allocator_type std::list< _Tp, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

insert() [1/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::list< _Tp, _Alloc >::insert (
    const_iterator __p,
    initializer_list< value_type > __l ) [inline]
```

Inserts the contents of an initializer_list into list before specified const_iterator.

Parameters

<code>__p</code>	A const_iterator into the list.
<code>__l</code>	An initializer_list of value_type.

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert copies of the data in the initializer_list / into the list before the location specified by *p*. This operation is linear in the number of elements inserted and does not invalidate iterators and references. References [std::list< _Tp, _Alloc >::insert\(\)](#).

insert() [2/5]

```
template<typename _Tp , typename _Alloc >
template<typename _InputIterator , typename >
list< _Tp, _Alloc >::iterator list::insert (
    const_iterator __position,
    _InputIterator __first,
    _InputIterator __last )
```

Inserts a range into the list.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert copies of the data in the range [*first,last*) into the list before the location specified by *position*. This operation is linear in the number of elements inserted and does not invalidate iterators and references. References [std::list< _Tp, _Alloc >::begin\(\)](#), and [std::list< _Tp, _Alloc >::empty\(\)](#).

insert() [3/5]

```
template<typename _Tp , typename _Alloc >
list< _Tp, _Alloc >::iterator list::insert (
    const_iterator __position,
    const value_type & __x )
```

Inserts given value into list before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references. Referenced by [std::list< _Tp, _Alloc >::insert\(\)](#).

insert() [4/5]

```
template<typename _Tp , typename _Alloc >
```

```
list<_Tp, _Alloc >::iterator list::insert (
    const_iterator __position,
    size_type __n,
    const value_type & __x )
```

Inserts a number of copies of given data into the list.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert a specified number of copies of the given data before the location specified by *position*. This operation is linear in the number of elements inserted and does not invalidate iterators and references. References [std::list<_Tp, _Alloc >::begin\(\)](#).

insert() [5/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
iterator std::list<_Tp, _Alloc >::insert (
    const_iterator __position,
    value_type && __x ) [inline]
```

Inserts given rvalue into list before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references. References [std::list<_Tp, _Alloc >::emplace\(\)](#), and [std::move\(\)](#).

max_size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
size_type std::list<_Tp, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the size() of the largest possible list.

References [__gnu_cxx::__alloc_traits<_Alloc, typename >::max_size\(\)](#).

merge() [1/2]

```
template<typename _Tp , typename _Alloc >
void list::merge (
    list<_Tp, _Alloc > && __x )
```

Merge sorted lists.

Parameters

<code>_↔</code>	Sorted list to merge.
<code>__x</code>	

Assumes that both `__x` and this list are sorted according to `operator<()`. Merges elements of `__x` into this list in sorted order, leaving `__x` empty when complete. Elements in this list precede elements in `__x` that are equal.

References [std::__addressof\(\)](#), [std::list<_Tp, _Alloc >::begin\(\)](#), [std::begin\(\)](#), [std::list<_Tp, _Alloc >::end\(\)](#), and [std::end\(\)](#).

merge() [2/2]

```
template<typename _Tp , typename _Alloc >
template<typename _StrictWeakOrdering >
void list::merge (
    list< _Tp, _Alloc > && __x,
    _StrictWeakOrdering __comp )
```

Merge sorted lists according to comparison function.

Template Parameters

<code>_StrictWeakOrdering</code>	Comparison function defining sort order.
--	--

Parameters

<code>__x</code>	Sorted list to merge.
<code>__comp</code>	Comparison functor.

Assumes that both `__x` and this list are sorted according to `StrictWeakOrdering`. Merges elements of `__x` into this list in sorted order, leaving `__x` empty when complete. Elements in this list precede elements in `__x` that are equivalent according to `StrictWeakOrdering()`.

References [std::__addressof\(\)](#), [std::list<_Tp, _Alloc >::begin\(\)](#), [std::begin\(\)](#), [std::list<_Tp, _Alloc >::end\(\)](#), and [std::end\(\)](#).

operator=() [1/3]

```
template<typename _Tp , typename _Alloc >
list< _Tp, _Alloc > & list::operator= (
    const list< _Tp, _Alloc > & __x )
```

List assignment operator.

Parameters

<code>_↔</code>	A list of identical element and allocator types.
<code>__x</code>	

All the elements of `__x` are copied.

Whether the allocator is copied depends on the allocator traits.

References [std::__addressof\(\)](#), [std::list<_Tp, _Alloc >::begin\(\)](#), and [std::list<_Tp, _Alloc >::end\(\)](#).

operator=() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
list & std::list< _Tp, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

List initializer list assignment operator.

Parameters

↔	An initializer_list of value_type.
↔	
↔	
↔	
/	

Replace the contents of the list with copies of the elements in the initializer_list __l. This is linear in l.size().
References [std::list< _Tp, _Alloc >::assign\(\)](#).

operator=() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
list & std::list< _Tp, _Alloc >::operator= (
    list< _Tp, _Alloc > && __x ) [inline], [noexcept]
```

List move assignment operator.

Parameters

↔	A list of identical element and allocator types.
__x	

The contents of __x are moved into this list (without copying).
Afterwards __x is a valid, but unspecified list
Whether the allocator is moved depends on the allocator traits.
References [std::move\(\)](#).

pop_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::pop_back ( ) [inline], [noexcept]
```

Removes last element.

This is a typical stack operation. It shrinks the list by one. Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the last element's data is needed, it should be retrieved before pop_back() is called.

pop_front()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::pop_front ( ) [inline], [noexcept]
```

Removes first element.

This is a typical stack operation. It shrinks the list by one. Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before pop_front() is called.


References [std::list< _Tp, _Alloc >::begin\(\)](#).

push_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::push_back (
    const value_type & __x ) [inline]
```

Add data to the end of the list.

Parameters

	Data to be added.
<code>__x</code>	


This is a typical stack operation. The function creates an element at the end of the list and assigns the given data to it. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references. References [std::list<_Tp, _Alloc>::end\(\)](#).

push_front()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::push_front (
    const value_type & __x ) [inline]
```

Add data to the front of the list.

Parameters

	Data to be added.
<code>__x</code>	

This is a typical stack operation. The function creates an element at the front of the list and assigns the given data to it. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references. References [std::list<_Tp, _Alloc>::begin\(\)](#).

rbegin() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::list<_Tp, _Alloc>::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the list. Iteration is done in reverse element order.

References [std::list<_Tp, _Alloc>::end\(\)](#).

rbegin() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reverse_iterator std::list<_Tp, _Alloc>::rbegin ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last element in the list. Iteration is done in reverse element order. References [std::list<_Tp, _Alloc>::end\(\)](#).

remove()

```
template<typename _Tp , typename _Alloc >
list<_Tp, _Alloc>::__remove_return_type list::remove (
    const _Tp & __value )
```

Remove all elements equal to value.

Parameters

<code>__value</code>	The value to remove.
----------------------	----------------------

Removes every element in the list equal to *value*. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::list< _Tp, _Alloc >::begin\(\)](#), [std::begin\(\)](#), [std::end\(\)](#), [std::list< _Tp, _Alloc >::size\(\)](#), and [std::list< _Tp, _Alloc >::splice\(\)](#).

remove_if()

```
template<typename _Tp , typename _Alloc >
template<typename _Predicate >
list< _Tp, _Alloc >::__remove_return_type list::remove_if (
    _Predicate __pred )
```

Remove all elements satisfying a predicate.

Template Parameters

<code>_Predicate</code>	Unary predicate function or object.
-------------------------	-------------------------------------

Removes every element in the list for which the predicate returns true. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::list< _Tp, _Alloc >::begin\(\)](#), [std::begin\(\)](#), [std::end\(\)](#), [std::list< _Tp, _Alloc >::size\(\)](#), and [std::list< _Tp, _Alloc >::splice\(\)](#).

rend() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
const_reverse_iterator std::list< _Tp, _Alloc >::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

References [std::list< _Tp, _Alloc >::begin\(\)](#).

rend() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
reverse_iterator std::list< _Tp, _Alloc >::rend ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

References [std::list< _Tp, _Alloc >::begin\(\)](#).

resize() [1/2]

```
template<typename _Tp , typename _Alloc >
void list::resize (
    size_type __new_size )
```

Resizes the list to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the list should contain.
-------------------------	---

This function will resize the list to the specified number of elements. If the number is smaller than the list's current size

the list is truncated, otherwise default constructed elements are appended.

References [std::end\(\)](#).

resize() [2/2]

```
template<typename _Tp , typename _Alloc >
void list::resize (
    size_type __new_size,
    const value_type & __x )
```

Resizes the list to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the list should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the list to the specified number of elements. If the number is smaller than the list's current size the list is truncated, otherwise the list is extended and new elements are populated with given data.

References [std::end\(\)](#).

reverse()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::reverse ( ) [inline], [noexcept]
```

Reverse the elements in list.

Reverse the order of elements in the list in linear time.

size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
size_type std::list<_Tp, _Alloc>::size ( ) const [inline], [noexcept]
```

Returns the number of elements in the list.

Referenced by [std::operator==\(\)](#), [std::list<_Tp, _Alloc>::remove\(\)](#), [std::list<_Tp, _Alloc>::remove_if\(\)](#), [std::list<_Tp, _Alloc>::unique](#) and [std::list<_Tp, _Alloc>::unique\(\)](#).

sort() [1/2]

```
template<typename _Tp , typename _Alloc >
void list::sort
```

Sort the elements.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

References [std::begin\(\)](#), [std::empty\(\)](#), and [std::end\(\)](#).

sort() [2/2]

```
template<typename _Tp , typename _Alloc >
template<typename _StrictWeakOrdering >
void list::sort (
    _StrictWeakOrdering __comp )
```

Sort the elements according to comparison function.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

References [std::begin\(\)](#), [std::empty\(\)](#), and [std::end\(\)](#).

splice() [1/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::splice (
    const_iterator __position,
    list< _Tp, _Alloc > && __x ) [inline], [noexcept]
```

Insert contents of another list.

Parameters

<code>__position</code>	Iterator referencing the element to insert before.
<code>__x</code>	Source list.

The elements of `__x` are inserted in constant time in front of the element referenced by `__position`. `__x` becomes an empty list.

Requires this `!= __x`.

References `std::list< _Tp, _Alloc >::begin()`, `std::list< _Tp, _Alloc >::empty()`, and `std::list< _Tp, _Alloc >::end()`.

Referenced by `std::list< _Tp, _Alloc >::remove()`, `std::list< _Tp, _Alloc >::remove_if()`, `std::list< _Tp, _Alloc >::splice()`, `std::list< _Tp, _Alloc >::splice()`, `std::list< _Tp, _Alloc >::unique()`, and `std::list< _Tp, _Alloc >::unique()`.

splice() [2/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::splice (
    const_iterator __position,
    list< _Tp, _Alloc > && __x,
    const_iterator __first,
    const_iterator __last ) [inline], [noexcept]
```

Insert range from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__first</code>	Const_iterator referencing the start of range in x.
<code>__last</code>	Const_iterator referencing the end of range in x.

Removes elements in the range `[__first, __last)` and inserts them before `__position` in constant time.

Undefined if `__position` is in `[__first, __last)`.

References `std::__addressof()`.

splice() [3/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list< _Tp, _Alloc >::splice (
    const_iterator __position,
    list< _Tp, _Alloc > && __x,
    const_iterator __i ) [inline], [noexcept]
```

Insert element from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.

Parameters

<code>__i</code>	Const_iterator referencing the element to move.
------------------	---

Removes the element in list `__x` referenced by `__i` and inserts it into the current list before `__position`.
References [std::__addressof\(\)](#).

splice() [4/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::splice (
    const_iterator __position,
    list<_Tp, _Alloc> & __x,
    const_iterator __first,
    const_iterator __last ) [inline], [noexcept]
```

Insert range from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__first</code>	Const_iterator referencing the start of range in x.
<code>__last</code>	Const_iterator referencing the end of range in x.

Removes elements in the range `[__first, __last)` and inserts them before `__position` in constant time.
Undefined if `__position` is in `[__first, __last)`.
References [std::move\(\)](#), and [std::list<_Tp, _Alloc>::splice\(\)](#).

splice() [5/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::splice (
    const_iterator __position,
    list<_Tp, _Alloc> & __x,
    const_iterator __i ) [inline], [noexcept]
```

Insert element from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__i</code>	Const_iterator referencing the element to move.

Removes the element in list `__x` referenced by `__i` and inserts it into the current list before `__position`.
References [std::move\(\)](#), and [std::list<_Tp, _Alloc>::splice\(\)](#).

swap()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
void std::list<_Tp, _Alloc>::swap (
    list<_Tp, _Alloc> & __x ) [inline], [noexcept]
```

Swaps data with another list.

Parameters

<code>_↔_X</code>	A list of the same element and allocator types.
-------------------	---

This exchanges the elements between two lists in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(l1,l2)` will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

unique() [1/2]

```
template<typename _Tp , typename _Alloc >
list< _Tp, _Alloc >::__remove_return_type list::unique
```

Remove consecutive duplicate elements.

For each consecutive set of elements with the same value, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References `std::list< _Tp, _Alloc >::begin()`, `std::begin()`, `std::end()`, `std::list< _Tp, _Alloc >::size()`, and `std::list< _Tp, _Alloc >::splice()`.

unique() [2/2]

```
template<typename _Tp , typename _Alloc >
template<typename _BinaryPredicate >
list< _Tp, _Alloc >::__remove_return_type list::unique (
    _BinaryPredicate __binary_pred )
```

Remove consecutive elements satisfying a predicate.

Template Parameters

<code>_BinaryPredicate</code>	Binary predicate function or object.
-------------------------------	--------------------------------------

For each consecutive set of elements `[first,last)` that satisfy `predicate(first,i)` where `i` is an iterator in `[first,last)`, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References `std::list< _Tp, _Alloc >::begin()`, `std::begin()`, `std::end()`, `std::list< _Tp, _Alloc >::size()`, and `std::list< _Tp, _Alloc >::splice()`.

The documentation for this class was generated from the following files:

- [stl_list.h](#)
- [list.tcc](#)

6.673 `__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >` Class

Template Reference

```
#include <assoc_container.hpp>
```

Inherits `detail::container_base_dispatch::type`.

Public Types

- typedef `list_update_tag` `container_category`
- typedef `Eq_Fn` `eq_fn`
- typedef `Update_Policy` `update_policy`

Public Member Functions

- **list_update** (const [list_update](#) &other)
- template<typename It >
 [list_update](#) (It first, It last)
- **list_update** & **operator=** (const [list_update](#) &other)
- void **swap** ([list_update](#) &other)

6.673.1 Detailed Description

template<typename Key, typename Mapped, class Eq_Fn = typename detail::default_eq_fn<Key>::type, class Update_Policy = detail::default_update_policy::type, class _Alloc = std::allocator<char>>
class __gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >

A list-update based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Eq_Fn</i>	Equal functor.
<i>Update_Policy</i>	Update policy, determines when an element will be moved to the front of the list.
<i>_Alloc</i>	Allocator type.

Base is detail::lu_map.

6.673.2 Constructor & Destructor Documentation**list_update()**

```
template<typename Key , typename Mapped , class Eq_Fn = typename detail::default_eq_fn<Key>↵
::type, class Update_Policy = detail::default_update_policy::type, class _Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >::list_update (
    It first,
    It last ) [inline]
```

Constructor taking __iterators to a range of value_types. The value_types between first_it and last_it will be inserted into the container object.

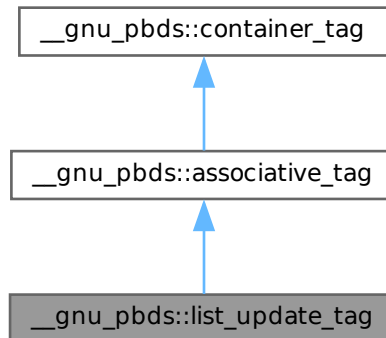
The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.674 __gnu_pbds::list_update_tag Struct Reference

```
#include <tag_and_trait.hpp>
```


Inheritance diagram for `__gnu_pbds::list_update_tag`:



6.674.1 Detailed Description

List-update.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.675 std::locale Class Reference

```
#include <locale_classes.h>
```

Classes

- class [facet](#)
- class [id](#)

Public Types

- typedef int [category](#)

Public Member Functions

- [locale](#) () throw ()
- [locale](#) (const char *__s)
- [locale](#) (const [locale](#) &__base, const char *__s, [category](#) __cat)
- [locale](#) (const [locale](#) &__base, const [locale](#) &__add, [category](#) __cat)
- [locale](#) (const [locale](#) &__base, const [std::string](#) &__s, [category](#) __cat)
- [locale](#) (const [locale](#) &__other) throw ()
- template<typename _Facet >
 [locale](#) (const [locale](#) &__other, _Facet *__f)
- [locale](#) (const [std::string](#) &__s)
- [~locale](#) () throw ()

- template<typename _Facet >
 locale combine (const **locale** &__other) const
- **string name** () const
- template<typename _Char , typename _Traits , typename _Alloc >
 bool **operator()** (const **basic_string**< _Char, _Traits, _Alloc > &__s1, const **basic_string**< _Char, _Traits, _Alloc > &__s2) const
- template<typename _CharT , typename _Traits , typename _Alloc >
 bool **operator()** (const **basic_string**< _CharT, _Traits, _Alloc > &__s1, const **basic_string**< _CharT, _Traits, _Alloc > &__s2) const
- const **locale** & **operator=** (const **locale** &__other) throw ()
- bool **operator==** (const **locale** &__other) const throw ()

Static Public Member Functions

- static const **locale** & **classic** ()
- static **locale global** (const **locale** &__loc)

Static Public Attributes

- static const **category none**
- static const **category ctype**
- static const **category numeric**
- static const **category collate**
- static const **category time**
- static const **category monetary**
- static const **category messages**
- static const **category all**

Friends

- template<typename _Cache >
 struct **__use_cache**
- class **_Impl**
- class **facet**
- template<typename _Facet >
 bool **has_facet** (const **locale** &) throw ()
- template<typename _Facet >
 const _Facet & **use_facet** (const **locale** &)

6.675.1 Detailed Description

Container class for localization functionality.

The locale class is first a class wrapper for C library locales. It is also an extensible container for user-defined localization. A locale is a collection of facets that implement various localization features such as money, time, and number printing. Constructing C++ locales does not change the C library locale.

This library supports efficient construction and copying of locales through a reference counting implementation of the locale class.

6.675.2 Member Typedef Documentation

category

```
typedef int std::locale::category
```

Definition of locale::category.

6.675.3 Constructor & Destructor Documentation

locale() [1/8]

```
std::locale::locale ( ) throw ( )
```

Default constructor.

Constructs a copy of the global locale. If no locale has been explicitly set, this is the C locale.

Referenced by [combine\(\)](#).

locale() [2/8]

```
std::locale::locale (
    const locale & __other ) throw ( )
```

Copy constructor.

Constructs a copy of *other*.

Parameters

<code>__other</code>	The locale to copy.
----------------------	---------------------

locale() [3/8]

```
std::locale::locale (
    const char * __s ) [explicit]
```

Named locale constructor.

Constructs a copy of the named C library locale.

Parameters

<code>__s</code>	Name of the locale to construct.
------------------	----------------------------------

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is null or an undefined locale.
---------------------------------	---

locale() [4/8]

```
std::locale::locale (
    const locale & __base,
    const char * __s,
    category __cat )
```

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale named by *s*. If *base* is named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__s</code>	Name of the locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from <code>__s</code> .

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is null or an undefined locale.
---------------------------------	---

locale() [5/8]

```
std::locale::locale (
    const std::string & __s ) [inline], [explicit]
```

Named locale constructor.

Constructs a copy of the named C library locale.

Parameters

<code>__s</code>	Name of the locale to construct.
------------------	----------------------------------

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is an undefined locale.
---------------------------------	---

locale() [6/8]

```
std::locale::locale (
    const locale & __base,
    const std::string & __s,
    category __cat ) [inline]
```

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale named by *s*. If *base* is named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__s</code>	Name of the locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from <code>__s</code> .

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is an undefined locale.
---------------------------------	---

locale() [7/8]

```
std::locale::locale (
    const locale & __base,
    const locale & __add,
    category __cat )
```

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale *add*. If *base* and *add* are named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__add</code>	The locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from add.

locale() [8/8]

```
template<typename _Facet >
std::locale::locale (
    const locale & __other,
    _Facet * __f )
```

Construct locale with another facet.

Constructs a copy of the locale `__other`. The facet `__f` is added to `__other`, replacing an existing facet of type `Facet` if there is one. If `__f` is null, this locale is a copy of `__other`.

Parameters

<code>__other</code>	The locale to copy.
<code>__f</code>	The facet to add in.

~locale()

```
std::locale::~~locale ( ) throw ( )
```

Locale destructor.

6.675.4 Member Function Documentation**classic()**

```
static const locale & std::locale::classic ( ) [static]
```

Return reference to the C locale.

combine()

```
template<typename _Facet >
locale std::locale::combine (
    const locale & __other ) const
```

Construct locale with another facet.

Constructs and returns a new copy of this locale. Adds or replaces an existing facet of type `Facet` from the locale *other* into the new locale.

Template Parameters

<code>_Facet</code>	The facet type to copy from other
---------------------	-----------------------------------

Parameters

Parameters

<code>__other</code>	The locale to copy from.
----------------------	--------------------------

Returns

Newly constructed locale.

Exceptions

<code>std::runtime_error</code>	if <code>__other</code> has no facet of type <code>_Facet</code> .
---------------------------------	--

References [locale\(\)](#).

global()

```
static locale std::locale::global (
    const locale & __loc ) [static]
```

Set global locale.

This function sets the global locale to the argument and returns a copy of the previous global locale. If the argument has a name, it will also call `std::setlocale(LC_ALL, loc.name())`.

Parameters

<code>__loc</code>	The new locale to make global.
--------------------	--------------------------------

Returns

Copy of the old global locale.

name()

```
string std::locale::name ( ) const
```

Return locale name.

Returns

Locale name or "*" if unnamed.

operator>()()

```
template<typename _Char , typename _Traits , typename _Alloc >
bool std::locale::operator() (
    const basic_string< _Char, _Traits, _Alloc > & __s1,
    const basic_string< _Char, _Traits, _Alloc > & __s2 ) const
```

Compare two strings according to collate.

Template operator to compare two strings using the compare function of the collate facet in this locale. One use is to provide the locale to the sort function. For example, a vector `v` of strings could be sorted according to locale `loc` by doing:

```
std::sort(v.begin(), v.end(), loc);
```

Parameters

<code>__s1</code>	First string to compare.
<code>__s2</code>	Second string to compare.

Returns

True if `collate<_Char> facet` compares `__s1 < __s2`, else false.

operator=()

```
const locale & std::locale::operator= (
    const locale & __other ) throw ( )
```

Assignment operator.

Set this locale to be a copy of *other*.

Parameters

<code>__other</code>	The locale to copy.
----------------------	---------------------

Returns

A reference to this locale.

operator==()

```
bool std::locale::operator== (
    const locale & __other ) const throw ( )
```

Locale equality.

Parameters

<code>__other</code>	The locale to compare against.
----------------------	--------------------------------

Returns

True if *other* and this refer to the same locale instance, are copies, or have the same name. False otherwise.

References [operator==\(\)](#).

Referenced by [operator==\(\)](#).

6.675.5 Friends And Related Symbol Documentation

has_facet

```
template<typename _Facet >
bool has_facet (
    const locale & ) throw ( ) [friend]
```

Test for the presence of a facet.

`has_facet` tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

use_facet

```
template<typename _Facet >
const _Facet & use_facet (
    const locale & ) [friend]
```

Return a facet.

`use_facet` looks for and returns a reference to a facet of type `Facet` where `Facet` is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type `Facet`.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

Reference to facet of type `Facet`.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

6.675.6 Member Data Documentation**all**

```
const category std::locale::all [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category `all` is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

collate

```
const category std::locale::collate [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

ctype

```
const category std::locale::ctype [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

messages

```
const category std::locale::messages [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

monetary

```
const category std::locale::monetary [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

none

```
const category std::locale::none [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

numeric

```
const category std::locale::numeric [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

time

```
const category std::locale::time [static]
```

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

The documentation for this class was generated from the following files:

- [locale_classes.h](#)
- [locale_classes.tcc](#)

6.676 std::lock_guard< _Mutex > Class Template Reference

```
#include <std_mutex.h>
```

Public Types

- typedef _Mutex **mutex_type**

Public Member Functions

- **lock_guard** (const [lock_guard](#) &)=delete
- **lock_guard** (mutex_type &__m)
- **lock_guard** (mutex_type &__m, [adopt_lock_t](#)) noexcept
- **lock_guard** & **operator=** (const [lock_guard](#) &)=delete

6.676.1 Detailed Description

```
template<typename _Mutex>
class std::lock_guard< _Mutex >
```

A simple scoped lock type.

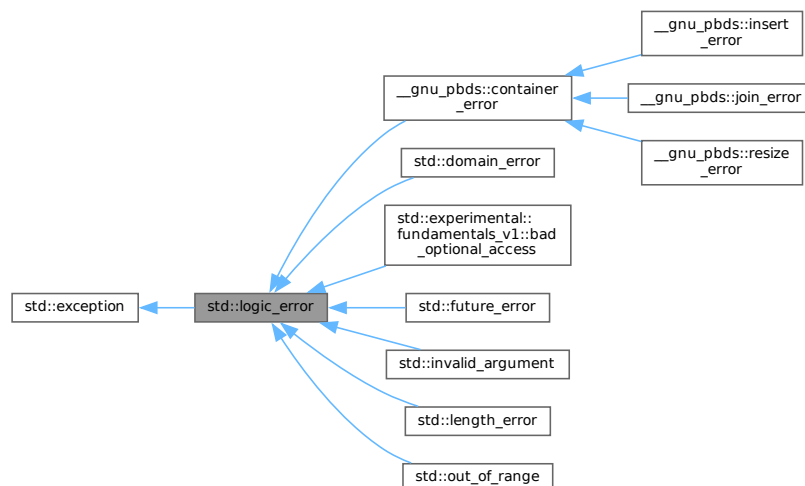
A lock_guard controls mutex ownership within a scope, releasing ownership in the destructor.

The documentation for this class was generated from the following file:

- [std_mutex.h](#)

6.677 std::logic_error Class Reference

Inheritance diagram for std::logic_error:



Public Member Functions

- **logic_error** (const char *) _GLIBCXX_TXN_SAFE
- **logic_error** (const [logic_error](#) &)=default
- **logic_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **logic_error** ([logic_error](#) &&) noexcept
- **logic_error** & **operator=** (const [logic_error](#) &)=default
- **logic_error** & **operator=** ([logic_error](#) &&) noexcept
- virtual const char * **what** () const noexcept

6.677.1 Detailed Description

One of two subclasses of exception.

Logic errors represent problems in the internal logic of a program; in theory, these are preventable, and even detectable before the program runs (e.g., violations of class invariants).

6.677.2 Constructor & Destructor Documentation

logic_error()

```
std::logic_error::logic_error (
    const string & __arg ) [explicit]
```

Takes a character string describing the error.

6.677.3 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

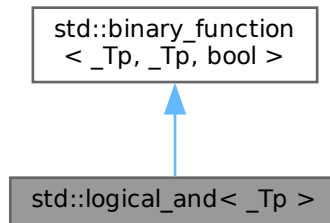
The documentation for this class was generated from the following file:

- [stdexcept](#)

6.678 std::logical_and<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::logical_and< _Tp >:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `bool` **operator()**(const `_Tp` &__x, const `_Tp` &__y) const

6.678.1 Detailed Description

```
template<typename _Tp>
struct std::logical_and< _Tp >
```

One of the [Boolean operations functors](#).

6.678.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary_function< _Tp , _Tp , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.679 `std::logical_and< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef `__is_transparent` `is_transparent`

Public Member Functions

- template<typename `_Tp`, typename `_Up` >
constexpr auto **operator()** (`_Tp` &&`_t`, `_Up` &&`_u`) const noexcept(noexcept(`std::forward`< `_Tp` >(`_t`) &&`std::forward`< `_Up` >(`_u`))) -> decltype(`std::forward`< `_Tp` >(`_t`) &&`std::forward`< `_Up` >(`_u`))

6.679.1 Detailed Description

One of the [Boolean operations functors](#).

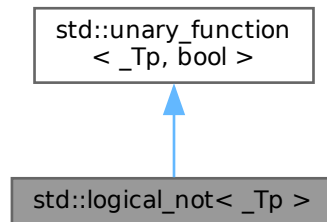
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.680 `std::logical_not< _Tp >` Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::logical_not< _Tp >`:



Public Types

- typedef `_Tp` `argument_type`
- typedef `bool` `result_type`

Public Member Functions

- constexpr `bool` **operator()** (const `_Tp` &`_x`) const

6.680.1 Detailed Description

```
template<typename _Tp>
struct std::logical_not< _Tp >
```

One of the [Boolean operations functors](#).

6.680.2 Member Typedef Documentation

argument_type

typedef _Tp [std::unary_function](#)< _Tp , bool >::argument_type [inherited]
 argument_type is the type of the argument

result_type

typedef bool [std::unary_function](#)< _Tp , bool >::result_type [inherited]
 result_type is the return type

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.681 std::logical_not< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent is **transparent**

Public Member Functions

- template<typename _Tp >
 constexpr auto **operator()** (_Tp &&__t) const noexcept(noexcept(![std::forward](#)< _Tp >(__t))) -> decltype(![std::forward](#)< _Tp >(__t))

6.681.1 Detailed Description

One of the [Boolean operations functors](#).

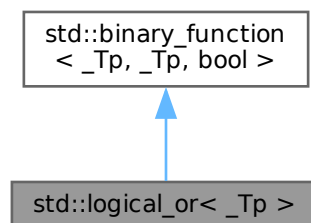
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.682 std::logical_or< _Tp > Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::logical_or< _Tp >:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `bool` **operator()** (`const _Tp &__x`, `const _Tp &__y`) `const`

6.682.1 Detailed Description

```
template<typename _Tp>
struct std::logical_or<_Tp>
```

One of the [Boolean operations functors](#).

6.682.2 Member Typedef Documentation

`first_argument_type`

```
typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

`result_type`

```
typedef bool std::binary_function<_Tp, _Tp, bool>::result_type [inherited]
result_type is the return type
```

`second_argument_type`

```
typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.683 `std::logical_or< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- template<typename `_Tp`, typename `_Up`>
constexpr auto **operator()** (`_Tp &&__t`, `_Up &&__u`) `const` noexcept(noexcept([std::forward](#)< `_Tp` >(__t)) || [std::forward](#)< `_Up` >(__u)) -> decltype([std::forward](#)< `_Tp` >(__t) || [std::forward](#)< `_Up` >(__u))

6.683.1 Detailed Description

One of the [Boolean operations functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.684 std::lognormal_distribution<_RealType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **lognormal_distribution** (_RealType __m, _RealType __s=_RealType(1))
- **lognormal_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **m** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()
- _RealType **s** () const

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::lognormal_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [lognormal_distribution](#) &__d1, const [lognormal_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::lognormal_distribution](#)< _RealType1 > &__x)

6.684.1 Detailed Description

template<typename _RealType = double>
class std::lognormal_distribution<_RealType>

A lognormal_distribution random number distribution.
The formula for the normal probability mass function is

$$p(x|m, s) = \frac{1}{sx\sqrt{2\pi}} \exp - \frac{(\ln x - m)^2}{2s^2}$$

6.684.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::lognormal_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.684.3 Member Function Documentation

max()

```
template<typename _RealType = double>
result_type std::lognormal_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits< _Tp >::max\(\)](#).

min()

```
template<typename _RealType = double>
result_type std::lognormal_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::lognormal_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::lognormal_distribution< _RealType >::operator>\(\)](#).

Referenced by [std::lognormal_distribution< _RealType >::operator>\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::lognormal_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::lognormal_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::lognormal_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

References [std::normal_distribution<_RealType>::reset\(\)](#).

6.684.4 Friends And Related Symbol Documentation

operator<<

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::lognormal_distribution< _RealType1 > & __x ) [friend]
Inserts a lognormal_distribution random number distribution __x into the output stream __os.
```

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A lognormal_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const lognormal_distribution< _RealType > & __d1,
    const lognormal_distribution< _RealType > & __d2 ) [friend]
```

Return true if two lognormal distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::lognormal_distribution< _RealType1 > & __x ) [friend]
Extracts a lognormal_distribution random number distribution __x from the input stream __is.
```

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A lognormal_distribution random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.685 `__gnu_pbds::detail::lu_counter_metadata< Size_Type >` Class Template Reference

```
#include <lu_counter_metadata.hpp>
```

Public Types

- typedef `Size_Type` **size_type**

Friends

- class `lu_counter_policy_base< size_type >`

6.685.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::detail::lu_counter_metadata< Size_Type >
```

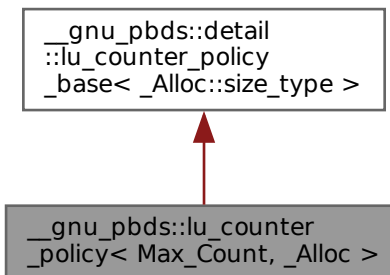
A list-update metadata type that moves elements to the front of the list based on the counter algorithm. The documentation for this class was generated from the following file:

- [lu_counter_metadata.hpp](#)

6.686 `__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >` Class Template Reference

```
#include <list_update_policy.hpp>
```

Inheritance diagram for `__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >`:



Public Types

- enum { `max_count` }
- typedef `_Alloc` **allocator_type**
- typedef `detail::rebind_traits< _Alloc, metadata_type >::reference` `metadata_reference`
- typedef `detail::lu_counter_metadata< size_type >` `metadata_type`
- typedef `allocator_type::size_type` **size_type**

Public Member Functions

- `metadata_type operator() ()` const
- `bool operator() (metadata_reference r_data)` const

Private Member Functions

- `bool operator() (Metadata_Reference r_data, size_type m_max_count) const`
- `lu_counter_metadata< size_type > operator() (size_type max_size) const`

6.686.1 Detailed Description

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
class __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >
```

A list-update policy that moves elements to the front of the list based on the counter algorithm.

6.686.2 Member Typedef Documentation

metadata_reference

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
typedef detail::rebind_traits<_Alloc, metadata_type>::reference __gnu_pbds::lu_counter_policy<
Max_Count, _Alloc >::metadata_reference
Reference to metadata on which this functor operates.
```

metadata_type

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
typedef detail::lu_counter_metadata<size_type> __gnu_pbds::lu_counter_policy< Max_Count, _Alloc
>::metadata_type
Metadata on which this functor operates.
```

6.686.3 Member Enumeration Documentation

anonymous enum

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
anonymous enum
```

Enumerator

max_count	When some element is accessed this number of times, it will be moved to the front of the list.
-----------	--

6.686.4 Member Function Documentation

operator>() [1/2]

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
metadata_type __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::operator() ( ) const [inline]
Creates a metadata object.
References __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::max_count.
```

operator>() [2/2]

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
bool __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::operator() (
    metadata_reference r_data ) const [inline]
```

Decides whether a metadata object should be moved to the front of the list.

References `__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::max_count`.

The documentation for this class was generated from the following file:

- [list_update_policy.hpp](#)

6.687 `__gnu_pbds::detail::lu_counter_policy_base< Size_Type >` Class Template Reference

```
#include <lu_counter_metadata.hpp>
```

Protected Types

- typedef `Size_Type` **size_type**

Protected Member Functions

- template<typename `Metadata_Reference` >
bool **operator()** (`Metadata_Reference` r_data, `size_type` m_max_count) const
- [lu_counter_metadata](#)< `size_type` > **operator()** (`size_type` max_size) const

6.687.1 Detailed Description

```
template<typename Size_Type>  
class __gnu_pbds::detail::lu_counter_policy_base< Size_Type >
```

Base class for list-update counter policy.

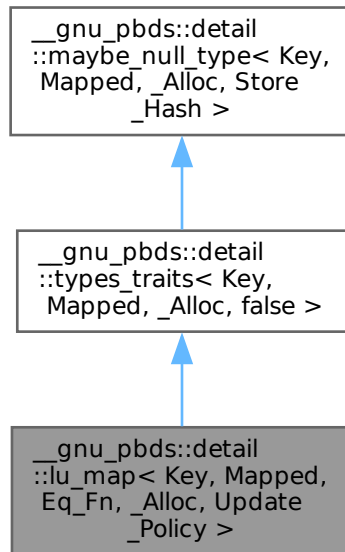
The documentation for this class was generated from the following file:

- [lu_counter_metadata.hpp](#)

6.688 `__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >` Class Template Reference

```
#include <lu_map_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `Eq_Fn` **eq_fn**
- typedef `iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**
- typedef `traits_base::key_const_reference` **key_const_reference**
- typedef `traits_base::key_pointer` **key_pointer**
- typedef `traits_base::key_reference` **key_reference**
- typedef `traits_base::key_type` **key_type**
- typedef `traits_base::mapped_const_pointer` **mapped_const_pointer**
- typedef `traits_base::mapped_const_reference` **mapped_const_reference**
- typedef `traits_base::mapped_pointer` **mapped_pointer**
- typedef `traits_base::mapped_reference` **mapped_reference**
- typedef `traits_base::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `point_const_iterator` **point_const_iterator**
- typedef `point_iterator` **point_iterator**
- typedef `traits_base::pointer` **pointer**
- typedef `traits_base::reference` **reference**

- typedef `_Alloc::size_type` **size_type**
- typedef `integral_constant< int, Store_Hash >` **store_extra**
- typedef `stored_data< value_type, size_type, Store_Hash >` **stored_data_type**
- typedef `Update_Policy::metadata_type` **update_metadata**
- typedef `Update_Policy` **update_policy**
- typedef `traits_base::value_type` **value_type**

Public Member Functions

- **lu_map** (const [lu_map](#)< Key, Mapped, Eq_Fn, _Alloc, Update_Policy > &)
- template<typename It >
 lu_map (It, It)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- template<typename Pred >
 size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference r_key)
- point_const_iterator **find** (key_const_reference r_key) const
- [std::pair](#)< point_iterator, bool > **insert** (const_reference)
- size_type **max_size** () const
- mapped_reference **operator[]** (key_const_reference r_key)
- size_type **size** () const
- void **swap** ([lu_map](#)< Key, Mapped, Eq_Fn, _Alloc, Update_Policy > &)

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Member Functions

- template<typename It >
 void **copy_from_range** (It, It)

Friends

- class **const_iterator_**
- class **iterator_**

6.688.1 Detailed Description

template<typename Key, typename Mapped, typename Eq_Fn, typename _Alloc, typename Update_Policy>
class [__gnu_pbds::detail::lu_map](#)< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >

list-based (with updates) associative container. Skip to the `lu`, my darling.

The documentation for this class was generated from the following file:

- [lu_map.hpp](#)

6.689 `__gnu_pbds::lu_move_to_front_policy<_Alloc>` Class Template Reference

```
#include <list_update_policy.hpp>
```

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `detail::rebind_traits<_Alloc, metadata_type>::reference` **metadata_reference**
- typedef `null_type` **metadata_type**

Public Member Functions

- `metadata_type operator() () const`
- `bool operator() (metadata_reference r_metadata) const`

6.689.1 Detailed Description

```
template<typename _Alloc = std::allocator<char>>
class __gnu_pbds::lu_move_to_front_policy<_Alloc>
```

A list-update policy that unconditionally moves elements to the front of the list. A null type means that each link in a list-based container does not actually need metadata.

6.689.2 Member Typedef Documentation**metadata_reference**

```
template<typename _Alloc = std::allocator<char>>
typedef detail::rebind_traits<_Alloc, metadata_type>::reference __gnu_pbds::lu_move_to_front_policy<
_Alloc>::metadata_reference
```

Reference to metadata on which this functor operates.

metadata_type

```
template<typename _Alloc = std::allocator<char>>
typedef null_type __gnu_pbds::lu_move_to_front_policy<_Alloc>::metadata_type
```

Metadata on which this functor operates.

6.689.3 Member Function Documentation**operator>() [1/2]**

```
template<typename _Alloc = std::allocator<char>>
metadata_type __gnu_pbds::lu_move_to_front_policy<_Alloc>::operator() ( ) const [inline]
```

Creates a metadata object.

operator>() [2/2]

```
template<typename _Alloc = std::allocator<char>>
bool __gnu_pbds::lu_move_to_front_policy<_Alloc>::operator() (
    metadata_reference r_metadata ) const [inline]
```

Decides whether a metadata object should be moved to the front of the list.

The documentation for this class was generated from the following file:

- `list_update_policy.hpp`

6.690 `std::make_signed<_Tp>` Struct Template Reference

Public Types

- `typedef __make_signed_selector<_Tp>::__type type`

6.690.1 Detailed Description

```
template<typename _Tp>
struct std::make_signed<_Tp>
```

`make_signed`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.691 `std::make_unsigned<_Tp>` Struct Template Reference

Public Types

- `typedef __make_unsigned_selector<_Tp>::__type type`

6.691.1 Detailed Description

```
template<typename _Tp>
struct std::make_unsigned<_Tp>
```

`make_unsigned`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.692 `__gnu_cxx::malloc_allocator<_Tp>` Class Template Reference

```
#include <malloc_allocator.h>
```

Public Types

- `typedef std::ptrdiff_t difference_type`
- `typedef std::true_type propagate_on_container_move_assignment`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `constexpr malloc_allocator (const malloc_allocator &) noexcept`
- `template<typename _Tp1 >
constexpr malloc_allocator (const malloc_allocator<_Tp1> &) noexcept`
- `_Tp * allocate (size_type __n, const void * = 0)`
- `void deallocate (_Tp * __p, size_type)`

Friends

- `template<typename _Up >
constexpr bool operator== (const malloc_allocator &, const malloc_allocator<_Up> &) noexcept`

6.692.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::malloc_allocator< _Tp >
```

An allocator that uses malloc.

This is precisely the allocator defined in the C++ Standard.

- all allocation calls malloc
- all deallocation calls free

The documentation for this class was generated from the following file:

- [malloc_allocator.h](#)

6.693 std::__debug::map< _Key, _Tp, _Compare, _Allocator > Class Template Reference

```
#include <map.h>
```

Inheritance diagram for std::__debug::map< _Key, _Tp, _Compare, _Allocator >:



Public Types

- typedef _Allocator **allocator_type**
- typedef __gnu_debug::Safe_iterator< _Base_const_iterator, map > **const_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef std::reverse_iterator< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- using insert_return_type = _Node_insert_return< iterator, node_type >
- typedef __gnu_debug::Safe_iterator< _Base_iterator, map > **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- typedef _Tp **mapped_type**
- using node_type = typename _Base::node_type
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef std::reverse_iterator< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef std::pair< const_Key, _Tp > **value_type**

Public Member Functions

- **map** ([_Base_ref](#) __x)
- `template<typename _InputIterator >`
map ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp=[_Compare](#)(), const [_Allocator](#) &__a=[_Allocator](#)())
- `template<typename _InputIterator >`
map ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- **map** (const [_Compare](#) &__comp, const [_Allocator](#) &__a=[_Allocator](#)())
- **map** (const [allocator_type](#) &__a)
- **map** (const [map](#) &)=default
- **map** (const [map](#) &__m, const [__type_identity_t](#)< [allocator_type](#) > &__a)
- **map** ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__c=[_Compare](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- **map** ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- **map** ([map](#) &&)=default
- **map** ([map](#) &&__m, const [__type_identity_t](#)< [allocator_type](#) > &__a) noexcept(noexcept([_Base](#)([std::move](#)(__m), __a)))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) **begin** () const noexcept
- [iterator](#) **begin** () noexcept
- [const_iterator](#) **cbegin** () const noexcept
- [const_iterator](#) **cend** () const noexcept
- void **clear** () noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- `template<typename... _Args>`
[std::pair](#)< [iterator](#), bool > **emplace** ([_Args](#) &&... __args)
- `template<typename... _Args>`
[iterator](#) **emplace_hint** (const [iterator](#) __pos, [_Args](#) &&... __args)
- [const_iterator](#) **end** () const noexcept
- [iterator](#) **end** () noexcept
- `template<typename _Kt, typename _Req = typename __has_is_transparent< _Compare, _Kt>::type>`
[std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const [_Kt](#) &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent< _Compare, _Kt>::type>`
[std::pair](#)< const [iterator](#), const [iterator](#) > **equal_range** (const [_Kt](#) &__x) const
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const [key_type](#) &__x)
- [std::pair](#)< const [iterator](#), const [iterator](#) > **equal_range** (const [key_type](#) &__x) const
- [_Base_iterator](#) **erase** ([_Base_const_iterator](#) __position)
- [size_type](#) **erase** (const [key_type](#) &__x)
- [iterator](#) **erase** (const [iterator](#) __first, const [iterator](#) __last)
- [iterator](#) **erase** (const [iterator](#) __position)
- [_GLIBCXX_ABI_TAG_CXX11](#) [iterator](#) **erase** ([iterator](#) __position)
- [node_type](#) **extract** (const [key_type](#) &__key)
- [node_type](#) **extract** (const [iterator](#) __position)
- `template<typename _Kt, typename _Req = typename __has_is_transparent< _Compare, _Kt>::type>`
[iterator](#) **find** (const [_Kt](#) &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent< _Compare, _Kt>::type>`
[const_iterator](#) **find** (const [_Kt](#) &__x) const
- [iterator](#) **find** (const [key_type](#) &__x)
- [const_iterator](#) **find** (const [key_type](#) &__x) const

- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
std::pair< **iterator**, bool > **insert** (_Pair &&__x)
- **std::pair**< **iterator**, bool > **insert** (const **value_type** &__x)
- **iterator** **insert** (const **iterator** __hint, node_type &&__nh)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator **insert** (const **iterator** __position, _Pair &&__x)
- **iterator** **insert** (const **iterator** __position, const **value_type** &__x)
- **iterator** **insert** (const **iterator** __position, **value_type** &&__x)
- **insert_return_type** **insert** (node_type &&__nh)
- void **insert** (std::initializer_list< **value_type** > __list)
- **std::pair**< **iterator**, bool > **insert** (**value_type** &&__x)
- template<typename _Obj >
std::pair< **iterator**, bool > **insert_or_assign** (const key_type &__k, _Obj &&__obj)
- template<typename _Obj >
iterator **insert_or_assign** (const **iterator** __hint, const key_type &__k, _Obj &&__obj)
- template<typename _Obj >
iterator **insert_or_assign** (const **iterator** __hint, key_type &&__k, _Obj &&__obj)
- template<typename _Obj >
std::pair< **iterator**, bool > **insert_or_assign** (key_type &&__k, _Obj &&__obj)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator **lower_bound** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator **lower_bound** (const _Kt &__x) const
- **iterator** **lower_bound** (const key_type &__x)
- **const_iterator** **lower_bound** (const key_type &__x) const
- **map** & **operator=** (const **map** &)=default
- **map** & **operator=** (initializer_list< **value_type** > __l)
- **map** & **operator=** (**map** &&)=default
- **const_reverse_iterator** **rbegin** () const noexcept
- **reverse_iterator** **rbegin** () noexcept
- **const_reverse_iterator** **rend** () const noexcept
- **reverse_iterator** **rend** () noexcept
- void **swap** (**map** &__x) noexcept(*/*conditional */*)
- template<typename... _Args>
pair< **iterator**, bool > **try_emplace** (const key_type &__k, _Args &&... __args)
- template<typename... _Args>
iterator **try_emplace** (const **iterator** __hint, const key_type &__k, _Args &&... __args)
- template<typename... _Args>
iterator **try_emplace** (const **iterator** __hint, key_type &&__k, _Args &&... __args)
- template<typename... _Args>
pair< **iterator**, bool > **try_emplace** (key_type &&__k, _Args &&... __args)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator **upper_bound** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator **upper_bound** (const _Kt &__x) const
- **iterator** **upper_bound** (const key_type &__x)
- **const_iterator** **upper_bound** (const key_type &__x) const

Protected Member Functions

- void **_M_swap** (_Safe_container &__x) noexcept

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >`
`class ::__gnu_debug::__Safe_iterator`

6.693.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator =
std::allocator<std::pair<const _Key, _Tp> >>
class std::__debug::map< _Key, _Tp, _Compare, _Allocator >
```

Class `std::map` with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [map.h](#)

6.694 `std::map< _Key, _Tp, _Compare, _Alloc >` Class Template Reference

```
#include <stl_map.h>
```

Public Types

- `typedef _Alloc allocator_type`
- `typedef _Rep_type::const_iterator const_iterator`
- `typedef _Alloc_traits::const_pointer const_pointer`
- `typedef _Alloc_traits::const_reference const_reference`
- `typedef _Rep_type::const_reverse_iterator const_reverse_iterator`
- `typedef _Rep_type::difference_type difference_type`
- `using insert_return_type = typename _Rep_type::insert_return_type`
- `typedef _Rep_type::iterator iterator`
- `typedef _Compare key_compare`
- `typedef _Key key_type`
- `typedef _Tp mapped_type`
- `using node_type = typename _Rep_type::node_type`
- `typedef _Alloc_traits::pointer pointer`
- `typedef _Alloc_traits::reference reference`
- `typedef _Rep_type::reverse_iterator reverse_iterator`
- `typedef _Rep_type::size_type size_type`
- `typedef std::pair< const _Key, _Tp > value_type`

Public Member Functions

- `map ()=default`
- `template<typename _InputIterator >`
`map (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator >`
`map (_InputIterator __first, _InputIterator __last, const _Compare &__comp, const allocator_type &__a=allocator_type())`
- `template<typename _InputIterator >`
`map (_InputIterator __first, _InputIterator __last, const allocator_type &__a)`
- `map (const _Compare &__comp, const allocator_type &__a=allocator_type())`
- `map (const allocator_type &__a)`
- `map (const map &)=default`
- `map (const map &__m, const __type_identity_t< allocator_type > &__a)`

- `map` (`initializer_list`< `value_type` > `__l`, `const _Compare &__comp=_Compare()`, `const allocator_type &__a=allocator_type()`)
- `map` (`initializer_list`< `value_type` > `__l`, `const allocator_type &__a`)
- `map` (`map` &&)=default
- `map` (`map` && `__m`, `const __type_identity_t< allocator_type > &__a`) noexcept(`is_nothrow_copy_constructible< _Compare >::value && _Alloc_traits::_S_always_equal()`)
- `~map` ()=default
- `mapped_type` & `at` (`const key_type &__k`)
- `const mapped_type` & `at` (`const key_type &__k`) const
- `const_iterator` `begin` () const noexcept
- `iterator` `begin` () noexcept
- `const_iterator` `cbegin` () const noexcept
- `const_iterator` `cend` () const noexcept
- `void` `clear` () noexcept
- `const_reverse_iterator` `crbegin` () const noexcept
- `const_reverse_iterator` `crend` () const noexcept
- `template<typename... _Args>`
`std::pair`< `iterator`, `bool` > `emplace` (`_Args` &&... `__args`)
- `template<typename... _Args>`
`iterator` `emplace_hint` (`const_iterator` `__pos`, `_Args` &&... `__args`)
- `bool` `empty` () const noexcept
- `const_iterator` `end` () const noexcept
- `iterator` `end` () noexcept
- `size_type` `erase` (`const key_type &__x`)
- `iterator` `erase` (`const_iterator` `__first`, `const_iterator` `__last`)
- `node_type` `extract` (`const key_type &__x`)
- `node_type` `extract` (`const_iterator` `__pos`)
- `allocator_type` `get_allocator` () const noexcept
- `template<typename _InputIterator >`
`void` `insert` (`_InputIterator` `__first`, `_InputIterator` `__last`)
- `iterator` `insert` (`const_iterator` `__hint`, `node_type` && `__nh`)
- `insert_return_type` `insert` (`node_type` && `__nh`)
- `void` `insert` (`std::initializer_list`< `value_type` > `__list`)
- `template<typename _Obj >`
`pair`< `iterator`, `bool` > `insert_or_assign` (`const key_type &__k`, `_Obj` && `__obj`)
- `template<typename _Obj >`
`iterator` `insert_or_assign` (`const_iterator` `__hint`, `const key_type &__k`, `_Obj` && `__obj`)
- `template<typename _Obj >`
`iterator` `insert_or_assign` (`const_iterator` `__hint`, `key_type` && `__k`, `_Obj` && `__obj`)
- `template<typename _Obj >`
`pair`< `iterator`, `bool` > `insert_or_assign` (`key_type` && `__k`, `_Obj` && `__obj`)
- `key_compare` `key_comp` () const
- `size_type` `max_size` () const noexcept
- `template<typename _Cmp2 >`
`void` `merge` (`map`< `_Key`, `_Tp`, `_Cmp2`, `_Alloc` > && `__source`)
- `template<typename _Cmp2 >`
`void` `merge` (`map`< `_Key`, `_Tp`, `_Cmp2`, `_Alloc` > & `__source`)
- `template<typename _Cmp2 >`
`void` `merge` (`multimap`< `_Key`, `_Tp`, `_Cmp2`, `_Alloc` > && `__source`)
- `template<typename _Cmp2 >`
`void` `merge` (`multimap`< `_Key`, `_Tp`, `_Cmp2`, `_Alloc` > & `__source`)

- `map & operator=` (const `map` &)=default
 - `map & operator=` (initializer_list< `value_type` > __l)
 - `map & operator=` (map &&)=default
 - mapped_type & `operator[]` (const key_type &__k)
 - mapped_type & `operator[]` (key_type &&__k)
 - `const_reverse_iterator rbegin` () const noexcept
 - `reverse_iterator rbegin` () noexcept
 - `const_reverse_iterator rend` () const noexcept
 - `reverse_iterator rend` () noexcept
 - size_type `size` () const noexcept
 - void `swap` (map &__x) noexcept(*conditional *)
 - template<typename... _Args>
pair< iterator, bool > `try_emplace` (const key_type &__k, _Args &&... __args)
 - template<typename... _Args>
iterator `try_emplace` (const_iterator __hint, const key_type &__k, _Args &&... __args)
 - template<typename... _Args>
iterator `try_emplace` (const_iterator __hint, key_type &&__k, _Args &&... __args)
 - template<typename... _Args>
pair< iterator, bool > `try_emplace` (key_type &&__k, _Args &&... __args)
 - value_compare `value_comp` () const
-
- `std::pair`< iterator, bool > `insert` (const `value_type` &__x)
 - `std::pair`< iterator, bool > `insert` (`value_type` &&__x)
 - template<typename _Pair >
__enable_if_t< `is_constructible`< `value_type`, _Pair >::value, pair< iterator, bool > > `insert` (_Pair &&__x)
-
- iterator `insert` (const_iterator __position, const `value_type` &__x)
 - iterator `insert` (const_iterator __position, `value_type` &&__x)
 - template<typename _Pair >
__enable_if_t< `is_constructible`< `value_type`, _Pair >::value, iterator > `insert` (const_iterator __position, _Pair &&__x)
-
- iterator `erase` (const_iterator __position)
 - _GLIBCXX_ABI_TAG_CXX11 iterator `erase` (iterator __position)
-
- iterator `find` (const key_type &__x)
 - template<typename _Kt >
auto `find` (const _Kt &__x) -> decltype(_M_t._M_find_tr(__x))
-
- const_iterator `find` (const key_type &__x) const
 - template<typename _Kt >
auto `find` (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x))
-
- size_type `count` (const key_type &__x) const

- template<typename _Kt >
auto **count** (const _Kt &__x) const -> decltype(_M_t._M_count_tr(__x))
- bool **contains** (const key_type &__x) const
- template<typename _Kt >
auto **contains** (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x), void(), true)
- iterator **lower_bound** (const key_type &__x)
- template<typename _Kt >
auto **lower_bound** (const _Kt &__x) -> decltype(iterator(_M_t._M_lower_bound_tr(__x)))
- const_iterator **lower_bound** (const key_type &__x) const
- template<typename _Kt >
auto **lower_bound** (const _Kt &__x) const -> decltype(const_iterator(_M_t._M_lower_bound_tr(__x)))
- iterator **upper_bound** (const key_type &__x)
- template<typename _Kt >
auto **upper_bound** (const _Kt &__x) -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))
- const_iterator **upper_bound** (const key_type &__x) const
- template<typename _Kt >
auto **upper_bound** (const _Kt &__x) const -> decltype(const_iterator(_M_t._M_upper_bound_tr(__x)))
- std::pair< iterator, iterator > **equal_range** (const key_type &__x)
- template<typename _Kt >
auto **equal_range** (const _Kt &__x) -> decltype(pair< iterator, iterator >(_M_t._M_equal_range_tr(__x)))
- std::pair< const_iterator, const_iterator > **equal_range** (const key_type &__x) const
- template<typename _Kt >
auto **equal_range** (const _Kt &__x) const -> decltype(pair< const_iterator, const_iterator >(_M_t._M_equal_range_tr(__x)))

Friends

- template<typename _K1, typename _T1, typename _C1, typename _A1 >
bool **operator**< (const map< _K1, _T1, _C1, _A1 > &, const map< _K1, _T1, _C1, _A1 > &)
- template<typename _K1, typename _T1, typename _C1, typename _A1 >
bool **operator**== (const map< _K1, _T1, _C1, _A1 > &, const map< _K1, _T1, _C1, _A1 > &)
- template<typename, typename >
struct **std::_Rb_tree_merge_helper**

6.694.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp> >>
class std::map< _Key, _Tp, _Compare, _Alloc >
```

A standard container made up of (key,value) pairs, which can be retrieved based on a key, in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using unique keys). For a `map<Key, T>` the `key_type` is `Key`, the `mapped_type` is `T`, and the `value_type` is `std::pair<const Key, T>`.

Maps support bidirectional iterators.

The private tree data is declared exactly the same way for `map` and `multimap`; the distinction is made entirely in how the tree functions are called (`*_unique` versus `*_equal`, same as the standard).

6.694.2 Constructor & Destructor Documentation

map() [1/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::map< _Key, _Tp, _Compare, _Alloc >::map ( ) [default]
```

Default constructor creates no elements.

map() [2/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Creates a map with no elements.

Parameters

<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

map() [3/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    const map< _Key, _Tp, _Compare, _Alloc > & ) [default]
```

Map copy constructor.

Whether the allocator is copied depends on the allocator traits.

map() [4/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    map< _Key, _Tp, _Compare, _Alloc > && ) [default]
```

Map move constructor.

The newly-created map contains the exact contents of the moved instance. The moved instance is a valid, but unspecified, map.

map() [5/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::map<_Key, _Tp, _Compare, _Alloc >::map (
    initializer_list< value_type > __l,
    const _Compare & __comp = _Compare(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a map from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

Create a map consisting of copies of the elements in the initializer_list `__l`. This is linear in N if the range is already sorted, and NlogN otherwise (where N is `__l.size()`).

map() [6/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::map<_Key, _Tp, _Compare, _Alloc >::map (
    const allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

map() [7/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::map<_Key, _Tp, _Compare, _Alloc >::map (
    const map<_Key, _Tp, _Compare, _Alloc > & __m,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Allocator-extended copy constructor.

map() [8/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::map<_Key, _Tp, _Compare, _Alloc >::map (
    map<_Key, _Tp, _Compare, _Alloc > && __m,
    const __type_identity_t< allocator_type > & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

map() [9/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::map<_Key, _Tp, _Compare, _Alloc >::map (
    initializer_list< value_type > __l,
    const allocator_type & __a ) [inline]
```

Allocator-extended initialier-list constructor.

map() [10/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a ) [inline]
```

Allocator-extended range constructor.

map() [11/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Builds a map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a map consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

map() [12/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::map< _Key, _Tp, _Compare, _Alloc >::map (
    _InputIterator __first,
    _InputIterator __last,
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a map consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

~map()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
std::map< _Key, _Tp, _Compare, _Alloc >::~~map ( ) [default]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

6.694.3 Member Function Documentation**at()**

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
mapped_type & std::map< _Key, _Tp, _Compare, _Alloc >::at (
    const key_type & __k ) [inline]
```

Access to map data.

Parameters

<code>__k</code>	The key for which data should be retrieved.
------------------	---

Returns

A reference to the data whose key is equivalent to `__k`, if such a data is present in the map.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

References [std::map< _Key, _Tp, _Compare, _Alloc >::end\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::key_comp\(\)](#), and [std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound\(\)](#).

begin() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

begin() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
iterator std::map< _Key, _Tp, _Compare, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

cbegin()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::map<_Key, _Tp, _Compare, _Alloc>::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

cend()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::map<_Key, _Tp, _Compare, _Alloc>::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the map. Iteration is done in ascending order according to the keys.

clear()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
void std::map<_Key, _Tp, _Compare, _Alloc>::clear ( ) [inline], [noexcept]
```

Erases all elements in a map. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
template<typename _Kt >
```

```
auto std::map<_Key, _Tp, _Compare, _Alloc>::contains (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x), void()), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

True if there is an element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
bool std::map<_Key, _Tp, _Compare, _Alloc>::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns


True if there is an element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map<_Key, _Tp, _Compare, _Alloc>::count (
    const _Kt & __x ) const -> decltype(_M_t._M_count_tr(__x))    [inline]
```

Finds the number of elements with given key.

Parameters

 _X	Key of (key, value) pairs to be located.
---	--

Returns

Number of elements with specified key.


This function only makes sense for multimaps; for map the result will either be 0 (not present) or 1 (present).

count() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::map<_Key, _Tp, _Compare, _Alloc>::count (
    const key_type & __x ) const    [inline]
```

Finds the number of elements with given key.

Parameters

 _X	Key of (key, value) pairs to be located.
---	--

Returns

Number of elements with specified key.

This function only makes sense for multimaps; for map the result will either be 0 (not present) or 1 (present).

crbegin()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc>::crbegin ( ) const    [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

crend()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
```

```
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

emplace()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename... _Args>
std::pair< iterator, bool > std::map< _Key, _Tp, _Compare, _Alloc >::emplace (
    _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the map.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

References `std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint()`, `std::map< _Key, _Tp, _Compare, _Alloc >::end()`, `std::map< _Key, _Tp, _Compare, _Alloc >::key_comp()`, and `std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound()`.

emplace_hint()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename... _Args>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the map.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the `std::pair` built from `__args` (may or may not be that `std::pair`).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

Referenced by [std::map< _Key, _Tp, _Compare, _Alloc >::emplace\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace\(\)](#), and [std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace\(\)](#).

empty()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
bool std::map< _Key, _Tp, _Compare, _Alloc >::empty ( ) const [inline], [noexcept]
Returns true if the map is empty. (Thus begin() would equal end().)
```

end() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::end ( ) const [inline], [noexcept]
Returns a read-only (constant) iterator that points one past the last pair in the map. Iteration is done in ascending order
according to the keys.
```

end() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::end ( ) [inline], [noexcept]
Returns a read/write iterator that points one past the last pair in the map. Iteration is done in ascending order according
to the keys.
Referenced by std::map< \_Key, \_Tp, \_Compare, \_Alloc >::at\(\), std::map< \_Key, \_Tp, \_Compare, \_Alloc >::emplace\(\),
std::map< \_Key, \_Tp, \_Compare, \_Alloc >::extract\(\), std::map< \_Key, \_Tp, \_Compare, \_Alloc >::insert\(\), std::map< \_Key, \_Tp, \_Compare, \_Alloc >::insert\_or\_assign\(\),
std::map< \_Key, \_Tp, \_Compare, \_Alloc >::operator\[\]\(\), and std::map< \_Key, \_Tp, \_Compare, \_Alloc >::try\_emplace\(\).
```

equal_range() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(pair<iterator, iterator>(_M_t._M_equal_range_tr(__x)))
[inline]
Finds a subsequence matching given key.
```

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

equal_range() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(pair<const_iterator, const_iterator> ( _M_t._M_↵
equal_range_tr(__x))) [inline]
Finds a subsequence matching given key.
```

Parameters

↵ __x	Key of (key, value) pairs to be located.
------------------------	--

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

equal_range() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::pair< iterator, iterator > std::map< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const key_type & __x ) [inline]
Finds a subsequence matching given key.
```

Parameters

↵ __x	Key of (key, value) pairs to be located.
------------------------	--

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

equal_range() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::pair< const_iterator, const_iterator > std::map< _Key, _Tp, _Compare, _Alloc >::equal_range
(
    const key_type & __x ) const [inline]
Finds a subsequence matching given key.
```

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

erase() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::map<_Key, _Tp, _Compare, _Alloc>::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map<_Key, _Tp, _Compare, _Alloc>::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [first,last) range of elements from a map.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map<_Key, _Tp, _Compare, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from a map.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
_GLIBCXX_ABI_TAG_CXX11 iterator std::map<_Key, _Tp, _Compare, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from a map.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
node_type std::map<_Key, _Tp, _Compare, _Alloc >::extract (
    const key_type & __x ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
node_type std::map<_Key, _Tp, _Compare, _Alloc>::extract (
    const_iterator __pos ) [inline]
```

Extract a node.


References [std::map<_Key, _Tp, _Compare, _Alloc>::end\(\)](#).

find() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map<_Key, _Tp, _Compare, _Alloc>::find (
    const _Kt & __x ) -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a map.

Parameters

 <code>__x</code>	Key of (key, value) pair to be located.
--	---

Returns

Iterator pointing to sought-after element, or end() if not found.


This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map<_Key, _Tp, _Compare, _Alloc>::find (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a map.

Parameters

 <code>__x</code>	Key of (key, value) pair to be located.
--	---

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
```

```
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::find (
    const key_type & __x ) [inline]
```

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

get_allocator()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
allocator_type std::map< _Key, _Tp, _Compare, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

insert() [1/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
void std::map< _Key, _Tp, _Compare, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair >::value, pair< iterator, bool > > std::map<
_Key, _Tp, _Compare, _Alloc >::insert (
    _Pair && __x ) [inline]
```

Attempts to insert a std::pair into the map.

Parameters

<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

References [std::map<_Key, _Tp, _Compare, _Alloc>::emplace_hint\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::end\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::key_comp\(\)](#), and [std::map<_Key, _Tp, _Compare, _Alloc>::lower_bound\(\)](#).

insert() [3/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::pair< iterator, bool > std::map< _Key, _Tp, _Compare, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Attempts to insert a std::pair into the map.

Parameters

<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

insert() [4/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [5/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair >::value, iterator > std::map< _Key, _Tp, _↵
_Compare, _Alloc >::insert (
    const_iterator __position,
    _Pair && __x ) [inline]
```

Attempts to insert a `std::pair` into the map.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

insert() [6/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __position,
    const value_type & __x ) [inline]
```

Attempts to insert a `std::pair` into the map.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

insert() [7/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map<_Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __position,
    value_type && __x ) [inline]
```

Attempts to insert a `std::pair` into the map.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

References [std::move\(\)](#).

insert() [8/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
insert_return_type std::map<_Key, _Tp, _Compare, _Alloc >::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

Referenced by [std::map<_Key, _Tp, _Compare, _Alloc >::insert\(\)](#), and [std::map<_Key, _Tp, _Compare, _Alloc >::operator\[\]\(\)](#).

insert() [9/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
void std::map<_Key, _Tp, _Compare, _Alloc >::insert (
    std::initializer_list< value_type > __list ) [inline]
```

Attempts to insert a list of `std::pairs` into the map.

Parameters

<code>__list</code>	A <code>std::initializer_list<value_type></code> of pairs to be inserted.
---------------------	---

Complexity similar to that of the range constructor.

References [std::map<_Key, _Tp, _Compare, _Alloc >::insert\(\)](#).

insert() [10/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::pair< iterator, bool > std::map< _Key, _Tp, _Compare, _Alloc >::insert (
    value_type && __x ) [inline]
```

Attempts to insert a `std::pair` into the map.

Parameters

<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
------------------	---

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

References [std::move\(\)](#).

insert_or_assign() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Obj >
pair< iterator, bool > std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign (
    const key_type & __k,
    _Obj && __obj ) [inline]
```

Attempts to insert or assign a `std::pair` into the map.

Parameters

<code>__k</code>	Key to use for finding a possibly existing pair in the map.
<code>__obj</code>	Argument used to generate the .second for a pair instance.

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map. If the pair was already in the map, the .second of the pair is assigned from `__obj`.

Insertion requires logarithmic time.

References [std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::end\(\)](#), [std::forward_as_tuple\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::key_comp\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound\(\)](#), and [std::piecewise_construct](#).

insert_or_assign() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Obj >
iterator std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign (
    const_iterator __hint,
    const key_type & __k,
    _Obj && __obj ) [inline]
```

Attempts to insert or assign a std::pair into the map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__k</code>	Key to use for finding a possibly existing pair in the map.
<code>__obj</code>	Argument used to generate the .second for a pair instance.

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map. If the pair was already in the map, the .second of the pair is assigned from `__obj`.

Insertion requires logarithmic time.

References [std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint\(\)](#), [std::forward_as_tuple\(\)](#), and [std::piecewise_construct](#).

key_comp()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
key_compare std::map< _Key, _Tp, _Compare, _Alloc >::key_comp ( ) const [inline]
```

Returns the key comparison object out of which the map was constructed.

Referenced by [std::map< _Key, _Tp, _Compare, _Alloc >::at\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::emplace\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::operator\[\]\(\)](#), and [std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace\(\)](#).

lower_bound() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) -> decltype(iterator(_M.t._M_lower_bound_tr(__x))) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to first element equal to or greater than key, or end().


This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) const -> decltype(const_iterator(_M_t._M_lower_bound_tr(__x)))
[inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
<code>__x</code>	

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().


This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
<code>__x</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().


This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists. Referenced by [std::map< _Key, _Tp, _Compare, _Alloc >::at\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::emplace\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::insert_or_assign\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::operator\[\]\(\)](#), and [std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace\(\)](#).

lower_bound() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) const [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

 <code>__x</code>	Key of (key, value) pair to be located.
--	---

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

max_size()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::map< _Key, _Tp, _Compare, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the map.

operator=() [1/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
map & std::map< _Key, _Tp, _Compare, _Alloc >::operator= (
    const map< _Key, _Tp, _Compare, _Alloc > & ) [default]
```

Map assignment operator.

Whether the allocator is copied depends on the allocator traits.

operator=() [2/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
map & std::map< _Key, _Tp, _Compare, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Map list assignment operator.

Parameters

<code>↵</code>	An initializer_list.
<code>_↵</code>	
<code>↵</code>	
<code>_↵</code>	
<code>/</code>	

This function fills a map with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the map and that the resulting map's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
map & std::map< _Key, _Tp, _Compare, _Alloc >::operator= (
    map< _Key, _Tp, _Compare, _Alloc > && ) [default]
```

Move assignment operator.

operator[]()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
mapped_type & std::map< _Key, _Tp, _Compare, _Alloc >::operator[] (
    const key_type & __k ) [inline]
```

Subscript (`[]`) access to map data.

Parameters

<code>_↵</code> <code>_k</code>	The key for which data should be retrieved.
------------------------------------	---

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript (`[]`) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned.

Lookup requires logarithmic time.

References [std::map<_Key, _Tp, _Compare, _Alloc>::end\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::insert\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::key_comp\(\)](#), [std::map<_Key, _Tp, _Compare, _Alloc>::lower_bound\(\)](#), and [std::piecewise_construct](#).

rbegin() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc >::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

rbegin() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
```

```
std::allocator<std::pair<const _Key, _Tp> >>
```

```
reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc>::rbegin ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

rend() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc>::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

rend() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc>::rend ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

size()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
size_type std::map<_Key, _Tp, _Compare, _Alloc>::size ( ) const [inline], [noexcept]
```

Returns the size of the map.



swap()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
void std::map<_Key, _Tp, _Compare, _Alloc>::swap (
    map<_Key, _Tp, _Compare, _Alloc> &__x ) [inline], [noexcept]
```

Swaps data with another map.

Parameters

	A map of the same element and allocator types.
	

This exchanges the elements between two maps in constant time. (It is only swapping a pointer, an integer, and an instance of the Compare type (which itself is often stateless and empty), so it should be quite fast.) Note that the global std::swap() function is specialized such that std::swap(m1,m2) will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

try_emplace() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
template<typename... _Args>
```

```
pair< iterator, bool > std::map<_Key, _Tp, _Compare, _Alloc>::try_emplace (
    const key_type &__k,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the map.

Parameters

<code>__k</code>	Key to use for finding a possibly existing pair in the map.
<code>__args</code>	Arguments used to generate the <code>.second</code> for a new pair instance.

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map. If a pair is not inserted, this function has no effect.

Insertion requires logarithmic time.

References [std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::end\(\)](#), [std::forward_as_tuple\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::key_comp\(\)](#), [std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound\(\)](#) and [std::piecewise_construct](#).

try_emplace() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename... _Args>
iterator std::map< _Key, _Tp, _Compare, _Alloc >::try_emplace (
    const_iterator __hint,
    const key_type & __k,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__k</code>	Key to use for finding a possibly existing pair in the map.
<code>__args</code>	Arguments used to generate the <code>.second</code> for a new pair instance.

Returns

An iterator that points to the element with key of the `std::pair` built from `__args` (may or may not be that `std::pair`).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `try_emplace()` does. However, if insertion did not take place, this function has no effect. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

References [std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint\(\)](#), [std::forward_as_tuple\(\)](#), and [std::piecewise_construct](#).

upper_bound() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
template<typename _Kt >
auto std::map<_Key, _Tp, _Compare, _Alloc>::upper_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_upper_bound_tr(__x))) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::map<_Key, _Tp, _Compare, _Alloc>::upper_bound (
    const _Kt & __x ) const -> decltype(const_iterator(_M_t._M_upper_bound_tr(__x)))
[inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

upper_bound() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::map<_Key, _Tp, _Compare, _Alloc>::upper_bound (
    const key_type & __x ) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::upper_bound (
    const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

_X	Key of (key, value) pair to be located.
--------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

value_comp()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
value_compare std::map< _Key, _Tp, _Compare, _Alloc >::value_comp ( ) const [inline]
```

Returns a value comparison object, built from the key comparison object out of which the map was constructed.

The documentation for this class was generated from the following file:

- [stl_map.h](#)

6.695 std::mask_array<_Tp> Class Template Reference

```
#include <mask_array.h>
```

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- [mask_array](#) (const [mask_array](#) &)
- template<class `_Dom` >
void **operator%=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void **operator%=** (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator&=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void **operator&=** (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator*=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void **operator*=** (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator+=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void **operator+=** (const [valarray](#)< `_Tp` > &) const

- template<class _Dom >
void **operator-=** (const _Expr< _Dom, _Tp > &) const
- void **operator-=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator/=** (const _Expr< _Dom, _Tp > &) const
- void **operator/=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator<<=** (const _Expr< _Dom, _Tp > &) const
- void **operator<<=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator=** (const _Expr< _Dom, _Tp > &) const
- template<class _Ex >
void **operator=** (const _Expr< _Ex, _Tp > &__e) const
- void **operator=** (const _Tp &) const
- **mask_array** & **operator=** (const **mask_array** &)
- void **operator=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator>>=** (const _Expr< _Dom, _Tp > &) const
- void **operator>>=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator^=** (const _Expr< _Dom, _Tp > &) const
- void **operator^=** (const valarray< _Tp > &) const
- template<class _Dom >
void **operator|=** (const _Expr< _Dom, _Tp > &) const
- void **operator|=** (const valarray< _Tp > &) const

Friends

- class valarray< _Tp >

6.695.1 Detailed Description

template<class _Tp>
class std::mask_array< _Tp >

Reference to selected subset of an array.

A mask_array is a reference to the actual elements of an array specified by a bitmask in the form of an array of bool. The way to get a mask_array is to call operator[](valarray<bool>) on a valarray. The returned mask_array then permits carrying operations out on the referenced subset of elements in the original valarray.

For example, if a mask_array is obtained using the array (false, true, false, true) as an argument, the mask array has two elements referring to array[1] and array[3] in the underlying array.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

6.695.2 Member Function Documentation

operator%=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator%= (
    const valarray< _Tp > & ) const
```

Modulo slice elements by corresponding elements of *v*.

operator&=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator&= (
    const valarray< _Tp > & ) const
```

Logical and slice elements with corresponding elements of *v*.

operator*=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator*= (
    const valarray< _Tp > & ) const
```

Multiply slice elements by corresponding elements of *v*.

operator+=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator+= (
    const valarray< _Tp > & ) const
```

Add corresponding elements of *v* to slice elements.

operator-=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator-= (
    const valarray< _Tp > & ) const
```

Subtract corresponding elements of *v* from slice elements.

operator/=(

```
template<class _Tp >
void std::mask_array< _Tp >::operator/= (
    const valarray< _Tp > & ) const
```

Divide slice elements by corresponding elements of *v*.

operator<<=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator<<= (
    const valarray< _Tp > & ) const
```

Left shift slice elements by corresponding elements of *v*.

operator>>=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator>>= (
    const valarray< _Tp > & ) const
```

Right shift slice elements by corresponding elements of *v*.

operator^=()

```
template<class _Tp >
void std::mask_array< _Tp >::operator^= (
    const valarray< _Tp > & ) const
```

Logical xor slice elements with corresponding elements of *v*.

operator" |=()

```
template<class _Tp >
void std::mask\_array< _Tp >::operator|= (
    const valarray< _Tp > & ) const
```

Logical or slice elements with corresponding elements of *v*.

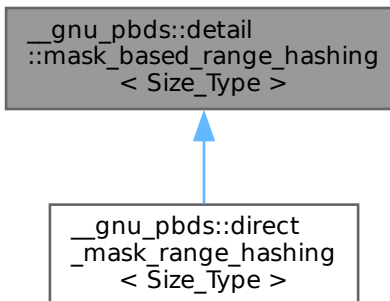
The documentation for this class was generated from the following files:

- [valarray](#)
- [mask_array.h](#)

6.696 `__gnu_pbds::detail::mask_based_range_hashing< Size_Type >` Class Template Reference

```
#include <mask_based_range_hashing.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::mask_based_range_hashing< Size_Type >`:



Protected Types

- typedef `Size_Type` **size_type**

Protected Member Functions

- void **notify_resized** (`size_type` size)
- `size_type` **range_hash** (`size_type` hash) const
- void **swap** ([mask_based_range_hashing](#) &other)

6.696.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::mask_based_range_hashing< Size_Type >
```

Range hashing policy.

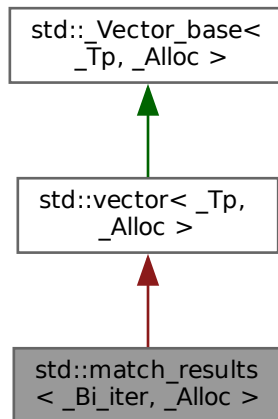
The documentation for this class was generated from the following file:

- [mask_based_range_hashing.hpp](#)

6.697 std::match_results< _Bi_iter, _Alloc > Class Template Reference

```
#include <regex.h>
```

Inheritance diagram for std::match_results< _Bi_iter, _Alloc >:



Public Types

28.10 Public Types

- typedef `sub_match< _Bi_iter >` **value_type**
- typedef const `value_type` & **const_reference**
- typedef `value_type` & **reference**
- typedef `_Base_type::const_iterator` **const_iterator**
- typedef `const_iterator` **iterator**
- typedef `__iter_traits::difference_type` **difference_type**
- typedef `allocator_traits< _Alloc >::size_type` **size_type**
- typedef `_Alloc` **allocator_type**
- typedef `__iter_traits::value_type` **char_type**
- typedef `std::basic_string< char_type >` **string_type**

Public Member Functions

- bool `ready` () const noexcept

28.10.1 Construction, Copying, and Destruction

- `match_results` ()
- `match_results` (const `_Alloc` & __a) noexcept
- `match_results` (const `match_results` &)=default
- `match_results` (`match_results` &&) noexcept=default
- `match_results` & `operator=` (const `match_results` &)=default
- `match_results` & `operator=` (`match_results` &&)=default
- `~match_results` ()=default

28.10.2 Size

- size_type [size](#) () const noexcept
- size_type [max_size](#) () const noexcept
- bool [empty](#) () const noexcept

28.10.4 Element Access

- difference_type [length](#) (size_type __sub=0) const
- difference_type [position](#) (size_type __sub=0) const
- string_type [str](#) (size_type __sub=0) const
- const_reference [operator\[\]](#) (size_type __sub) const
- const_reference [prefix](#) () const
- const_reference [suffix](#) () const
- const_iterator [begin](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [end](#) () const noexcept
- const_iterator [cend](#) () const noexcept

28.10.5 Formatting

These functions perform formatted substitution of the matched character sequences into their target. The format specifiers and escape sequences accepted by these functions are determined by their `flags` parameter as documented above.

- template<typename _Out_iter >
_Out_iter [format](#) (_Out_iter __out, const char_type *__fmt_first, const char_type *__fmt_last, [match_flag_type](#) __flags=[regex_constants::format_default](#)) const
- template<typename _Out_iter, typename _St, typename _Sa >
_Out_iter [format](#) (_Out_iter __out, const [basic_string](#)< char_type, _St, _Sa > &__fmt, [match_flag_type](#) __flags=[regex_constants::format_default](#)) const
- template<typename _St, typename _Sa >
[basic_string](#)< char_type, _St, _Sa > [format](#) (const [basic_string](#)< char_type, _St, _Sa > &__fmt, [match_flag_type](#) __flags=[regex_constants::format_default](#)) const
- string_type [format](#) (const char_type *__fmt, [match_flag_type](#) __flags=[regex_constants::format_default](#)) const

28.10.6 Allocator

- allocator_type [get_allocator](#) () const noexcept

28.10.7 Swap

- void [swap](#) ([match_results](#) &__that) noexcept

Private Member Functions

- constexpr iterator [begin](#) () noexcept
- constexpr iterator [end](#) () noexcept
- constexpr const_reference [operator\[\]](#) (size_type __n) const noexcept
- constexpr reference [operator\[\]](#) (size_type __n) noexcept
- constexpr void [swap](#) ([vector](#) &__x) noexcept

Friends

- template<typename, typename, typename >
class [regex_iterator](#)

6.697.1 Detailed Description

```
template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >>
class std::match_results< _Bi_iter, _Alloc >
```

The results of a match or search operation.

A collection of character sequences representing the result of a regular expression match. Storage for the collection is allocated and freed as necessary by the member functions of class template `match_results`.

This class satisfies the Sequence requirements, with the exception that only the operations defined for a const-qualified Sequence are supported.

The `sub_match` object stored at index 0 represents sub-expression 0, i.e. the whole match. In this case the `sub_match` member `matched` is always true. The `sub_match` object stored at index `n` denotes what matched the marked sub-expression `n` within the matched expression. If the sub-expression `n` participated in a regular expression match then the `sub_match` member `matched` evaluates to true, and members `first` and `second` denote the range of characters `[first, second)` which formed that match. Otherwise `matched` is false, and members `first` and `second` point to the end of the sequence that was searched.

6.697.2 Constructor & Destructor Documentation

`match_results()` [1/4]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
std::match_results< _Bi_iter, _Alloc >::match_results ( ) [inline]
```

Constructs a default `match_results` container.

Postcondition

`size()` returns 0 and `str()` returns an empty string.

`match_results()` [2/4]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
std::match_results< _Bi_iter, _Alloc >::match_results (
    const _Alloc & __a ) [inline], [explicit], [noexcept]
```

Constructs a default `match_results` container.

Postcondition

`size()` returns 0 and `str()` returns an empty string.

`match_results()` [3/4]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
std::match_results< _Bi_iter, _Alloc >::match_results (
    const match_results< _Bi_iter, _Alloc > & ) [default]
```

Copy constructs a `match_results`.

`match_results()` [4/4]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
std::match_results< _Bi_iter, _Alloc >::match_results (
    match_results< _Bi_iter, _Alloc > && ) [default], [noexcept]
```

Move constructs a `match_results`.

~match_results()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
std::match_results<_Bi_iter, _Alloc>::~~match_results ( ) [default]
```

Destroys a match_results object.

6.697.3 Member Function Documentation**begin()**

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_iterator std::match_results<_Bi_iter, _Alloc>::begin ( ) const [inline], [noexcept]
```

Gets an iterator to the start of the sub_match collection.

References [std::vector<_Tp, _Alloc>::begin\(\)](#).

Referenced by [std::match_results<_Bi_iter, _Alloc>::cbegin\(\)](#), and [std::operator==\(\)](#).

cbegin()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_iterator std::match_results<_Bi_iter, _Alloc>::cbegin ( ) const [inline], [noexcept]
```

Gets an iterator to the start of the sub_match collection.

References [std::match_results<_Bi_iter, _Alloc>::begin\(\)](#).

cend()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_iterator std::match_results<_Bi_iter, _Alloc>::cend ( ) const [inline], [noexcept]
```

Gets an iterator to one-past-the-end of the collection.

References [std::match_results<_Bi_iter, _Alloc>::end\(\)](#).

empty()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
bool std::match_results<_Bi_iter, _Alloc>::empty ( ) const [inline], [noexcept]
```

Indicates if the match_results contains no results.

Return values

<i>true</i>	The match_results object is empty.
<i>false</i>	The match_results object is not empty.

References [std::vector<_Tp, _Alloc>::size\(\)](#).

Referenced by [std::operator==\(\)](#), [std::match_results<_Bi_iter, _Alloc>::prefix\(\)](#), and [std::match_results<_Bi_iter, _Alloc>::suffix\(\)](#).

end()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_iterator std::match_results<_Bi_iter, _Alloc>::end ( ) const [inline], [noexcept]
```

Gets an iterator to one-past-the-end of the collection.

References [std::vector<_Tp, _Alloc>::empty\(\)](#), and [std::vector<_Tp, _Alloc>::end\(\)](#).

Referenced by [std::match_results<_Bi_iter, _Alloc>::cend\(\)](#), and [std::operator==\(\)](#).

format() [1/4]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
template<typename _Out_iter , typename _St , typename _Sa >
```



```

_Out_iter std::match_results< _Bi_iter, _Alloc >::format (
    _Out_iter __out,
    const basic_string< char_type, _St, _Sa > & __fmt,
    match_flag_type __flags = regex_constants::format_default ) const [inline]

```

Precondition

ready() == true

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::match_results< _Bi_iter, _Alloc >::format\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

format() [2/4]

```

template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
template<typename _Out_iter >
_Out_iter std::match_results< _Bi_iter, _Alloc >::format (
    _Out_iter __out,
    const char_type * __fmt_first,
    const char_type * __fmt_last,
    match_flag_type __flags = regex_constants::format_default ) const

```

Precondition

ready() == true

Referenced by [std::match_results< _Bi_iter, _Alloc >::format\(\)](#), [std::match_results< _Bi_iter, _Alloc >::format\(\)](#), and [std::match_results< _Bi_iter, _Alloc >::format\(\)](#).

format() [3/4]

```

template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
template<typename _St , typename _Sa >
basic_string< char_type, _St, _Sa > std::match_results< _Bi_iter, _Alloc >::format (
    const basic_string< char_type, _St, _Sa > & __fmt,
    match_flag_type __flags = regex_constants::format_default ) const [inline]

```

Precondition

ready() == true

References [std::back_inserter\(\)](#), and [std::match_results< _Bi_iter, _Alloc >::format\(\)](#).

format() [4/4]

```

template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
string_type std::match_results< _Bi_iter, _Alloc >::format (
    const char_type * __fmt,
    match_flag_type __flags = regex_constants::format_default ) const [inline]

```

Precondition

ready() == true

References [std::back_inserter\(\)](#), and [std::match_results< _Bi_iter, _Alloc >::format\(\)](#).

get_allocator()

```

template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
allocator_type std::match_results< _Bi_iter, _Alloc >::get_allocator ( ) const [inline], [noexcept]
Gets a copy of the allocator.

```

References [std::vector< _Tp, _Alloc >::get_allocator\(\)](#).

length()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
difference_type std::match_results<_Bi_iter, _Alloc>::length (
    size_type __sub = 0 ) const [inline]
```

Gets the length of the indicated submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function returns the length of the indicated submatch, or the length of the entire match if `__sub` is zero (the default).

max_size()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
size_type std::match_results<_Bi_iter, _Alloc>::max_size ( ) const [inline], [noexcept]
```

Gets the number of matches and submatches.

The number of matches for a given regular expression will be either 0 if there was no match or `mark_count() + 1` if a match was successful. Some matches may be empty.

Returns

the number of matches found.

References `std::vector<_Tp, _Alloc>::max_size()`.

operator=() [1/2]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
match_results & std::match_results<_Bi_iter, _Alloc>::operator= (
    const match_results<_Bi_iter, _Alloc> & ) [default]
```

Assigns rhs to *this.

operator=() [2/2]

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
match_results & std::match_results<_Bi_iter, _Alloc>::operator= (
    match_results<_Bi_iter, _Alloc> && ) [default]
```

Move-assigns rhs to *this.

operator[]()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_reference std::match_results<_Bi_iter, _Alloc>::operator[] (
    size_type __sub ) const [inline]
```

Gets a `sub_match` reference for the match or submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function gets a reference to the indicated submatch, or the entire match if `__sub` is zero.

If `__sub >= size()` then this function returns a `sub_match` with a special value indicating no submatch.

References [std::vector<_Tp, _Alloc>::operator\[\]\(\)](#), [std::match_results<_Bi_iter, _Alloc>::ready\(\)](#), and [std::match_results<_Bi_iter, _](#)

position()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
difference_type std::match_results<_Bi_iter, _Alloc>::position (
    size_type __sub = 0 ) const [inline]
```

Gets the offset of the beginning of the indicated submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function returns the offset from the beginning of the target sequence to the beginning of the submatch, unless the value of `__sub` is zero (the default), in which case this function returns the offset from the beginning of the target sequence to the beginning of the match.

References [std::distance\(\)](#).

prefix()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_reference std::match_results<_Bi_iter, _Alloc>::prefix ( ) const [inline]
```

Gets a `sub_match` representing the match prefix.

Precondition

`ready() == true`

This function gets a reference to a `sub_match` object representing the part of the target range between the start of the target range and the start of the match.

References [std::match_results<_Bi_iter, _Alloc>::empty\(\)](#), and [std::match_results<_Bi_iter, _Alloc>::ready\(\)](#).

Referenced by [std::operator==\(\)](#).

ready()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
bool std::match_results<_Bi_iter, _Alloc>::ready ( ) const [inline], [noexcept]
```

Indicates if the `match_results` is ready.

Return values

<code>true</code>	The object has a fully-established result state.
<code>false</code>	The object is not ready.

References [std::vector<_Tp, _Alloc>::empty\(\)](#).

Referenced by [std::operator==\(\)](#), [std::match_results<_Bi_iter, _Alloc>::operator\[\]\(\)](#), [std::match_results<_Bi_iter, _Alloc>::prefix\(\)](#),

and `std::match_results<_Bi_iter, _Alloc>::suffix()`.

size()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
size_type std::match_results<_Bi_iter, _Alloc>::size ( ) const [inline], [noexcept]
```

Gets the number of matches and submatches.

The number of matches for a given regular expression will be either 0 if there was no match or `mark_count() + 1` if a match was successful. Some matches may be empty.

Returns

the number of matches found.

References `std::vector<_Tp, _Alloc>::empty()`, and `std::vector<_Tp, _Alloc>::size()`.

Referenced by `std::operator==()`, and `std::match_results<_Bi_iter, _Alloc>::operator[]()`.

str()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
string_type std::match_results<_Bi_iter, _Alloc>::str (
    size_type __sub = 0 ) const [inline]
```

Gets the match or submatch converted to a string type.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function gets the submatch (or match, if `__sub` is zero) extracted from the target range and converted to the associated string type.

suffix()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
const_reference std::match_results<_Bi_iter, _Alloc>::suffix ( ) const [inline]
```

Gets a `sub_match` representing the match suffix.

Precondition

`ready() == true`

This function gets a reference to a `sub_match` object representing the part of the target range between the end of the match and the end of the target range.

References `std::match_results<_Bi_iter, _Alloc>::empty()`, and `std::match_results<_Bi_iter, _Alloc>::ready()`.

Referenced by `std::operator==()`.

swap()

```
template<typename _Bi_iter , typename _Alloc = allocator<sub_match<_Bi_iter> >>
void std::match_results<_Bi_iter, _Alloc>::swap (
    match_results<_Bi_iter, _Alloc> & __that ) [inline], [noexcept]
```

Swaps the contents of two `match_results`.

References `std::swap()`, `std::match_results<_Bi_iter, _Alloc>::swap()`, and `std::vector<_Tp, _Alloc>::swap()`.

Referenced by [std::match_results<_Bi_iter, _Alloc>::swap\(\)](#).

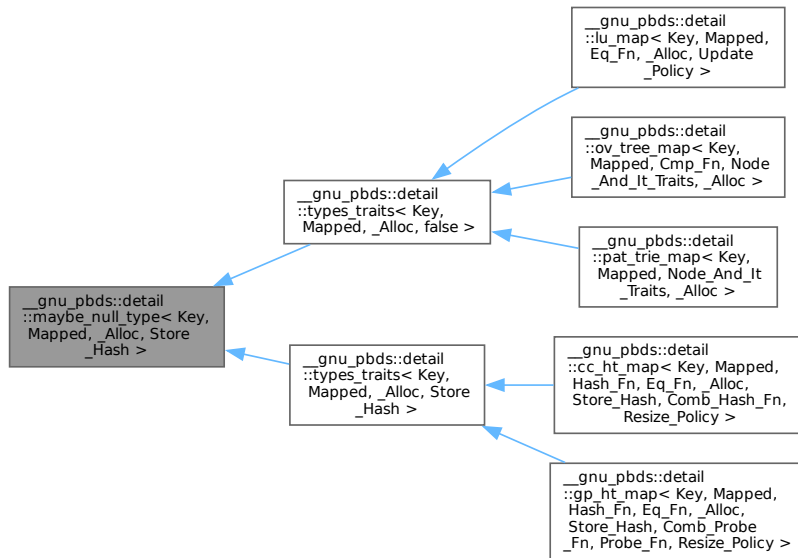
The documentation for this class was generated from the following file:

- [regex.h](#)

6.698 __gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference

```
#include <types_traits.hpp>
```

Inheritance diagram for __gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash >:



6.698.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash >
```

Base class for conditionally defining a static data member.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.699 __gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash > Struct Template Reference

```
#include <types_traits.hpp>
```

Static Public Attributes

- static `null_type` `s_null_type`

6.699.1 Detailed Description

```
template<typename Key, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash >
```

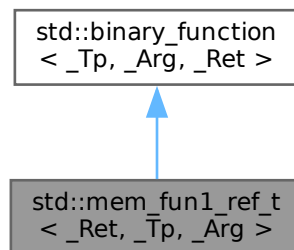
Specialization that defines a static data member of type null_type.
The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.700 std::mem_fun1_ref_t< _Ret, _Tp, _Arg > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::mem_fun1_ref_t< _Ret, _Tp, _Arg >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef _Ret [result_type](#)
- typedef _Arg [second_argument_type](#)

Public Member Functions

- **mem_fun1_ref_t** (_Ret(_Tp::*__pf)(_Arg))
- **operator()** (_Tp &__r, _Arg __x) const

6.700.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::mem_fun1_ref_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

6.700.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Arg , _Ret >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef _Ret std::binary\_function< _Tp , _Arg , _Ret >::result_type [inherited]
```

result_type is the return type

second_argument_type

```
typedef _Arg std::binary\_function< _Tp , _Arg , _Ret >::second_argument_type [inherited]
```

second_argument_type is the type of the second argument

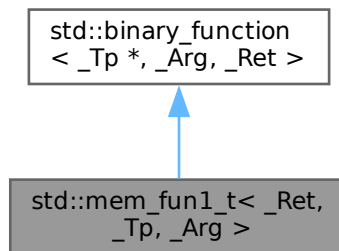
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.701 std::mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::mem_fun1_t< _Ret, _Tp, _Arg >:

**Public Types**

- typedef _Tp * [first_argument_type](#)
- typedef _Ret [result_type](#)
- typedef _Arg [second_argument_type](#)

Public Member Functions

- **mem_fun1_t** (_Ret(_Tp::* __pf)(_Arg))
- **operator()** (_Tp *__p, _Arg __x) const

6.701.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::mem_fun1_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

6.701.2 Member Typedef Documentation

first_argument_type

typedef _Tp * [std::binary_function](#)< _Tp * , _Arg , _Ret >::first_argument_type [inherited]
 first_argument_type is the type of the first argument

result_type

typedef _Ret [std::binary_function](#)< _Tp * , _Arg , _Ret >::result_type [inherited]
 result_type is the return type

second_argument_type

typedef _Arg [std::binary_function](#)< _Tp * , _Arg , _Ret >::second_argument_type [inherited]
 second_argument_type is the type of the second argument

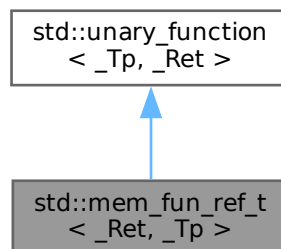
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.702 std::mem_fun_ref_t< _Ret, _Tp > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::mem_fun_ref_t< _Ret, _Tp >:



Public Types

- typedef _Tp [argument_type](#)
- typedef _Ret [result_type](#)

Public Member Functions

- `mem_fun_ref_t` (_Ret(_Tp::*__pf)())
- `_Ret operator()` (_Tp &__r) const

6.702.1 Detailed Description

`template<typename _Ret, typename _Tp>`
class `std::mem_fun_ref_t<_Ret, _Tp>`

One of the [adaptors for member pointers](#).

6.702.2 Member Typedef Documentation

argument_type

```
typedef _Tp std::unary_function< _Tp , _Ret >::argument_type [inherited]
```

argument_type is the type of the argument

result_type

```
typedef _Ret std::unary_function< _Tp , _Ret >::result_type [inherited]
```

result_type is the return type

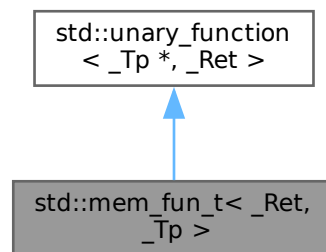
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.703 std::mem_fun_t< _Ret, _Tp > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::mem_fun_t< _Ret, _Tp >:



Public Types

- typedef _Tp * [argument_type](#)
- typedef _Ret [result_type](#)

Public Member Functions

- `mem_fun_t` (_Ret(_Tp::*__pf)())
- `_Ret operator()` (_Tp *__p) const

6.703.1 Detailed Description

```
template<typename _Ret, typename _Tp>
```

```
class std::mem_fun_t< _Ret, _Tp >
```

One of the [adaptors for member pointers](#).

6.703.2 Member Typedef Documentation

argument_type

typedef `_Tp *` [std::unary_function](#)< `_Tp *` , `_Ret` >::argument_type [inherited]

argument_type is the type of the argument

result_type

typedef `_Ret` [std::unary_function](#)< `_Tp *` , `_Ret` >::result_type [inherited]

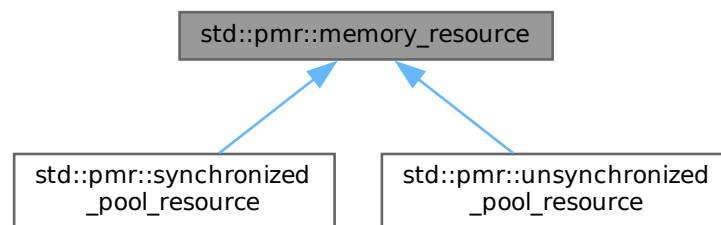
result_type is the return type

The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.704 std::pmr::memory_resource Class Reference

Inheritance diagram for std::pmr::memory_resource:



Public Member Functions

- **memory_resource** (const [memory_resource](#) &)=default
- void * **allocate** (size_t __bytes, size_t __alignment=_S_max_align)
- void **deallocate** (void * __p, size_t __bytes, size_t __alignment=_S_max_align)
- bool **is_equal** (const [memory_resource](#) & __other) const noexcept
- [memory_resource](#) & **operator=** (const [memory_resource](#) &)=default

6.704.1 Detailed Description

Class memory_resource.

The documentation for this class was generated from the following file:

- [memory_resource](#)

6.705 std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f> Class Template Reference

```
#include <random.h>
```

Public Types

- typedef `_UIntType` [result_type](#)

Public Member Functions

- template<typename `_Sseq`, typename `= _If_seed_seq<_Sseq>>`
[mersenne_twister_engine](#) (`_Sseq &__q`)
- [mersenne_twister_engine](#) ([result_type](#) `__sd`)
- void [discard](#) (unsigned long long `__z`)
- [result_type](#) [operator\(\)](#) ()
- template<typename `_Sseq` >
`_If_seed_seq<_Sseq>` [seed](#) (`_Sseq &__q`)
- template<typename `_Sseq` >
auto [seed](#) (`_Sseq &__q`) -> `_If_seed_seq<_Sseq>`
- void [seed](#) ([result_type](#) `__sd`=default_seed)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr [result_type](#) [default_seed](#)
- static constexpr [result_type](#) [initialization_multiplier](#)
- static constexpr `size_t` [mask_bits](#)
- static constexpr `size_t` [shift_size](#)
- static constexpr `size_t` [state_size](#)
- static constexpr [result_type](#) [tempering_b](#)
- static constexpr [result_type](#) [tempering_c](#)
- static constexpr [result_type](#) [tempering_d](#)
- static constexpr `size_t` [tempering_l](#)
- static constexpr `size_t` [tempering_s](#)
- static constexpr `size_t` [tempering_t](#)
- static constexpr `size_t` [tempering_u](#)
- static constexpr `size_t` [word_size](#)
- static constexpr [result_type](#) [xor_mask](#)

Friends

- template<typename `_UIntType1`, `size_t` `__w1`, `size_t` `__n1`, `size_t` `__m1`, `size_t` `__r1`, `_UIntType1` `__a1`, `size_t` `__u1`, `_UIntType1` `__d1`, `size_t` `__s1`, `_UIntType1` `__b1`, `size_t` `__t1`, `_UIntType1` `__c1`, `size_t` `__l1`, `_UIntType1` `__f1`, typename `_CharT`, typename `_Traits` >
[std::basic_ostream](#)< `_CharT`, `_Traits` > & [operator<<](#) ([std::basic_ostream](#)< `_CharT`, `_Traits` > & `__os`, const [std::mersenne_twister_engine](#)< `_UIntType1`, `__w1`, `__n1`, `__m1`, `__r1`, `__a1`, `__u1`, `__d1`, `__s1`, `__b1`, `__t1`, `__c1`, `__l1`, `__f1` > & `__x`)
- bool [operator==](#) (const [mersenne_twister_engine](#) & `__lhs`, const [mersenne_twister_engine](#) & `__rhs`)
- template<typename `_UIntType1`, `size_t` `__w1`, `size_t` `__n1`, `size_t` `__m1`, `size_t` `__r1`, `_UIntType1` `__a1`, `size_t` `__u1`, `_UIntType1` `__d1`, `size_t` `__s1`, `_UIntType1` `__b1`, `size_t` `__t1`, `_UIntType1` `__c1`, `size_t` `__l1`, `_UIntType1` `__f1`, typename `_CharT`, typename `_Traits` >
[std::basic_istream](#)< `_CharT`, `_Traits` > & [operator>>](#) ([std::basic_istream](#)< `_CharT`, `_Traits` > & `__is`, [std::mersenne_twister_engine](#)< `_UIntType1`, `__w1`, `__n1`, `__m1`, `__r1`, `__a1`, `__u1`, `__d1`, `__s1`, `__b1`, `__t1`, `__c1`, `__l1`, `__f1` > & `__x`)

6.705.1 Detailed Description

```
template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
class std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >
```

A generalized feedback shift register discrete random number generator.

This algorithm avoids multiplication and division and is designed to be friendly to a pipelined architecture. If the parameters are chosen correctly, this generator will produce numbers with a very long period and fairly good apparent entropy, although still not cryptographically strong.

The best way to use this generator is with the predefined `mt19937` class.

This algorithm was originally invented by Makoto Matsumoto and Takuji Nishimura.

Template Parameters

<code>__w</code>	Word size, the number of bits in each element of the state vector.
<code>__n</code>	The degree of recursion.
<code>__m</code>	The period parameter.
<code>__r</code>	The separation point bit index.
<code>__a</code>	The last row of the twist matrix.
<code>__u</code>	The first right-shift tempering matrix parameter.
<code>__d</code>	The first right-shift tempering matrix mask.
<code>__s</code>	The first left-shift tempering matrix parameter.
<code>__b</code>	The first left-shift tempering matrix mask.
<code>__t</code>	The second left-shift tempering matrix parameter.
<code>__c</code>	The second left-shift tempering matrix mask.
<code>__l</code>	The second right-shift tempering matrix parameter.
<code>__f</code>	Initialization multiplier.

6.705.2 Member Typedef Documentation

`result_type`

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, __f>
_UIntType
```

```
typedef _UIntType std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::result_type
```

The type of the generated random value.

6.705.3 Constructor & Destructor Documentation

mersenne_twister_engine()

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, ↵
__l, __f >::mersenne_twister_engine (
    _Sseq & __q ) [inline], [explicit]
```

Constructs a mersenne_twister_engine random number generator engine seeded from the seed sequence __q.

Parameters

<code>__q</code>	the seed sequence.
------------------	--------------------

6.705.4 Member Function Documentation

discard()

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
void std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, ↵
__c, __l, __f >::discard (
    unsigned long long __z )
```

Discard a sequence of random numbers.

max()

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
static constexpr result_type std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, ↵
__u, __d, __s, __b, __t, __c, __l, __f >::max ( ) [inline], [static], [constexpr]
```

Gets the largest possible value in the output range.

min()

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
static constexpr result_type std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, ↵
__u, __d, __s, __b, __t, __c, __l, __f >::min ( ) [inline], [static], [constexpr]
```

Gets the smallest possible value in the output range.

6.705.5 Friends And Related Symbol Documentation

operator<<

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a,
size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _↵
UIntType __f>
```

```
template<typename _UIntType1 , size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT , typename _Traits >
```

```
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::mersenne_twister_engine< _UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, __c1, __l1, __f1 > & __x ) [friend]
```

Inserts the current state of a % mersenne_twister_engine random number generator engine __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A % mersenne_twister_engine random number generator engine.

Returns

The output stream with the state of __x inserted or in an error state.

operator==

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
```

```
bool operator== (
    const mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > & __lhs,
    const mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > & __rhs ) [friend]
```

Compares two % mersenne_twister_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A % mersenne_twister_engine random number generator object.
<code>__rhs</code>	Another % mersenne_twister_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

operator>>

```
template<typename _UIntType , size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
```

```
template<typename _UIntType1 , size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT , typename _Traits >
```

```
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::mersenne_twister_engine< _UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, __c1, __l1, __f1 > & __x ) [friend]
```

Extracts the current state of a % mersenne_twister_engine random number generator engine `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A % mersenne_twister_engine random number generator engine.

Returns

The input stream with the state of `__x` extracted or in an error state.

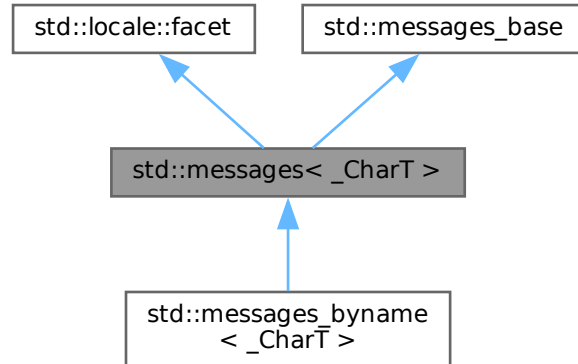
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.706 `std::messages<_CharT>` Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::messages<_CharT>`:



Public Types

- typedef int **catalog**
- typedef `_CharT` [char_type](#)
- typedef [basic_string<_CharT>](#) [string_type](#)

Public Member Functions

- `messages` (`__c_locale __cloc`, `const char *__s`, `size_t __refs=0`)
- `messages` (`size_t __refs=0`)
- `void close` (`catalog __c`) `const`
- `string_type get` (`catalog __c`, `int __set`, `int __msgid`, `const string_type &__s`) `const`
- `catalog open` (`const basic_string<char> &`, `const locale &`, `const char *`) `const`
- `catalog open` (`const basic_string<char> &__s`, `const locale &__loc`) `const`

Static Public Attributes

- static `locale::id id`

Protected Member Functions

- virtual `~messages` ()
- `string_type M_convert_from_char` (`char *`) `const`
- `char * M_convert_to_char` (`const string_type &__msg`) `const`
- virtual `void do_close` (`catalog`) `const`
- `void do_close` (`catalog`) `const`
- `void do_close` (`catalog`) `const`
- `string do_get` (`catalog`, `int`, `int`, `const string &`) `const`
- virtual `string_type do_get` (`catalog`, `int`, `int`, `const string_type &__default`) `const`
- `wstring do_get` (`catalog`, `int`, `int`, `const wstring &`) `const`
- virtual `catalog do_open` (`const basic_string<char> &`, `const locale &`) `const`
- `messages<char>::catalog do_open` (`const basic_string<char> &`, `const locale &`) `const`
- `messages<wchar_t>::catalog do_open` (`const basic_string<char> &`, `const locale &`) `const`

Static Protected Member Functions

- static `__c_locale S_clone_c_locale` (`__c_locale &__cloc`) `throw ()`
- static `void S_create_c_locale` (`__c_locale &__cloc`, `const char *__s`, `__c_locale __old=0`)
- static `void S_destroy_c_locale` (`__c_locale &__cloc`)
- static `__c_locale S_get_c_locale` ()
- static `const char * S_get_c_name` () `throw ()`
- static `__c_locale S_lc_ctype_c_locale` (`__c_locale __cloc`, `const char *__s`)

Protected Attributes

- `__c_locale M_c_locale_messages`
- `const char * M_name_messages`

6.706.1 Detailed Description

`template<typename _CharT>`
`class std::messages<_CharT>`

Primary class template `messages`.

This facet encapsulates the code to retrieve messages from message catalogs. The only thing defined by the standard for this facet is the interface. All underlying functionality is implementation-defined.

This library currently implements 3 versions of the message facet. The first version (gnu) is a wrapper around `gettext`, provided by `libintl`. The second version (ieee) is a wrapper around `catgets`. The final version (default) does no actual translation. These implementations are only provided for `char` and `wchar_t` instantiations.

The `messages` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `messages` facet.

6.706.2 Member Typedef Documentation

char_type

```
template<typename _CharT >
typedef _CharT std::messages< _CharT >::char_type
Public typedefs.
```

string_type

```
template<typename _CharT >
typedef basic_string<_CharT> std::messages< _CharT >::string_type
Public typedefs.
```

6.706.3 Constructor & Destructor Documentation

messages() [1/2]

```
template<typename _CharT >
std::messages< _CharT >::messages (
    size_t __refs = 0 ) [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

messages() [2/2]

```
template<typename _CharT >
std::messages< _CharT >::messages (
    __c_locale __cloc,
    const char * __s,
    size_t __refs = 0 ) [explicit]
```

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__s</code>	The name of a locale.
<code>__refs</code>	Refcount to pass to the base class.

~messages()

```
template<typename _CharT >
std::messages< _CharT >::~~messages [protected], [virtual]
Destructor.
```

6.706.4 Member Function Documentation

do_get()

```
string std::messages< char >::do_get (
    catalog ,
    int ,
    int ,
    const string & ) const [protected]
```

Specializations for required instantiations.

6.706.5 Member Data Documentation

id

```
template<typename _CharT >
locale::id std::messages< _CharT >::id [static]
```

Numpunct facet id.

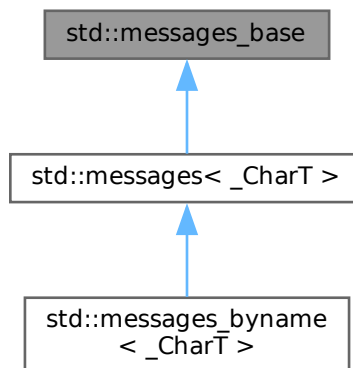
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [messages_members.h](#)

6.707 std::messages_base Struct Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::messages_base:



Public Types

- typedef int **catalog**

6.707.1 Detailed Description

Messages facet base class providing catalog typedef.

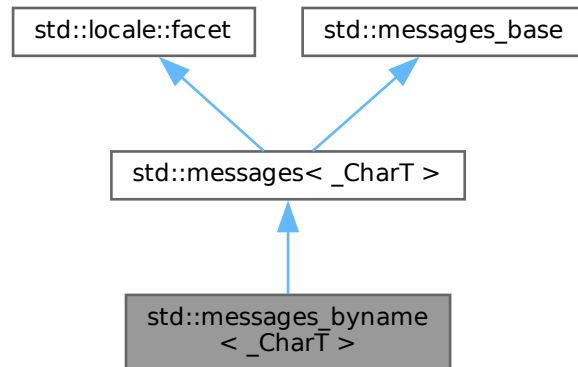
The documentation for this struct was generated from the following file:

- [locale_facets_nonio.h](#)

6.708 std::messages_byname<_CharT> Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::messages_byname<_CharT>:



Public Types

- typedef int **catalog**
- typedef `_CharT` **char_type**
- typedef [basic_string](#)<`_CharT`> **string_type**

Public Member Functions

- **messages_byname** (const char *__s, size_t __refs=0)
- **messages_byname** (const [string](#) &__s, size_t __refs=0)
- void **close** (catalog __c) const
- [string_type](#) **get** (catalog __c, int __set, int __msgid, const [string_type](#) &__s) const
- catalog **open** (const [basic_string](#)< char > &, const [locale](#) &, const char *) const
- catalog **open** (const [basic_string](#)< char > &__s, const [locale](#) &__loc) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- [string_type](#) **_M_convert_from_char** (char *) const
- char * **_M_convert_to_char** (const [string_type](#) &__msg) const
- virtual void **do_close** (catalog) const
- void **do_close** (catalog) const
- void **do_close** (catalog) const
- [string](#) **do_get** (catalog, int, int, const [string](#) &) const

- virtual [string_type](#) **do_get** (catalog, int, int, const [string_type](#) &__dfault) const
- [wstring](#) **do_get** (catalog, int, int, const [wstring](#) &) const
- virtual catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const
- [messages](#)< char >::catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const
- [messages](#)< wchar_t >::catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const

Static Protected Member Functions

- static [__c_locale](#) **_S_clone_c_locale** ([__c_locale](#) &__cloc) throw ()
- static void **_S_create_c_locale** ([__c_locale](#) &__cloc, const char *__s, [__c_locale](#) __old=0)
- static void **_S_destroy_c_locale** ([__c_locale](#) &__cloc)
- static [__c_locale](#) **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static [__c_locale](#) **_S_lc_type_c_locale** ([__c_locale](#) __cloc, const char *__s)

Protected Attributes

- [__c_locale](#) **_M_c_locale_messages**
- const char * **_M_name_messages**

6.708.1 Detailed Description

```
template<typename _CharT>
class std::messages_byname< _CharT >
```

class messages_byname [22.2.7.2].

6.708.2 Member Function Documentation

do_get()

```
string std::messages< char >::do_get (
    catalog ,
    int ,
    int ,
    const string & ) const [protected], [inherited]
```

Specializations for required instantiations.

6.708.3 Member Data Documentation

id

```
template<typename _CharT >
locale::id std::messages< _CharT >::id [static], [inherited]
```

Numpunct facet id.

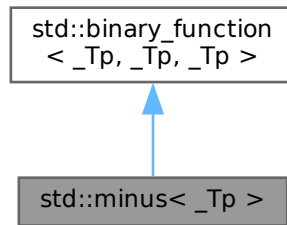
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [messages_members.h](#)

6.709 std::minus<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::minus<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `_Tp` **operator()** (const `_Tp` &__x, const `_Tp` &__y) const

6.709.1 Detailed Description

template<typename `_Tp`>
struct `std::minus<_Tp>`

One of the [math functors](#).

6.709.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::first_argument_type [inherited]
`first_argument_type` is the type of the first argument

result_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::result_type [inherited]
`result_type` is the return type

second_argument_type

typedef `_Tp` [std::binary_function](#)< `_Tp` , `_Tp` , `_Tp` >::second_argument_type [inherited]
`second_argument_type` is the type of the second argument

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.710 std::minus< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward< _Tp >(__t) - std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t) - std::forward< _Up >(__u))

6.710.1 Detailed Description

One of the [math functors](#).

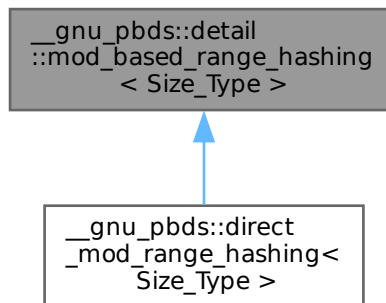
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.711 __gnu_pbds::detail::mod_based_range_hashing< Size_Type > Class Template Reference

```
#include <mod_based_range_hashing.hpp>
```

Inheritance diagram for __gnu_pbds::detail::mod_based_range_hashing< Size_Type >:



Protected Types

- typedef Size_Type **size_type**

Protected Member Functions

- void **notify_resized** (size_type s)
- size_type **range_hash** (size_type s) const
- void **swap** ([mod_based_range_hashing](#) &other)

6.711.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::mod_based_range_hashing< Size_Type >
```

Mod based range hashing.

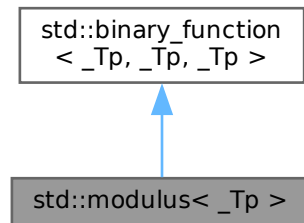
The documentation for this class was generated from the following file:

- [mod_based_range_hashing.hpp](#)

6.712 std::modulus<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::modulus<_Tp>:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `_Tp` **operator()** (const `_Tp` &__x, const `_Tp` &__y) const

6.712.1 Detailed Description

```
template<typename _Tp>
struct std::modulus<_Tp>
```

One of the [math functors](#).

6.712.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary\_function<_Tp, _Tp, _Tp>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

typedef `_Tp` `std::binary_function< _Tp , _Tp , _Tp >::result_type` [inherited]
`result_type` is the return type

second_argument_type

typedef `_Tp` `std::binary_function< _Tp , _Tp , _Tp >::second_argument_type` [inherited]
`second_argument_type` is the type of the second argument
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.713 `std::modulus< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- template<typename `_Tp` , typename `_Up` >
constexpr auto **operator()** (`_Tp` &&`_t`, `_Up` &&`_u`) const noexcept(noexcept(`std::forward< _Tp >(_t)` % `std::forward< _Up >(_u)`)) -> decltype(`std::forward< _Tp >(_t)` % `std::forward< _Up >(_u)`)

6.713.1 Detailed Description

One of the [math functors](#).

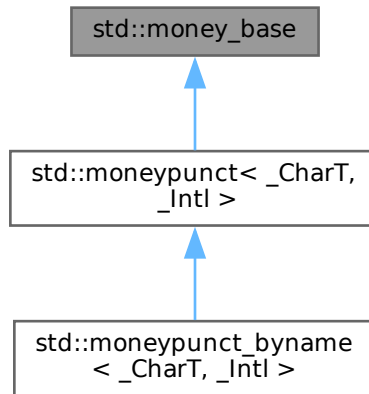
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.714 `std::money_base` Class Reference

```
#include <locale_facets_nonio.h>
```


Inheritance diagram for `std::money_base`:



Public Types

- enum { **_S_minus** , **_S_zero** , **_S_end** }
- enum **part** {
 none , **space** , **symbol** , **sign** ,
 value }

Static Public Member Functions

- static pattern **_S_construct_pattern** (char __precedes, char __space, char __posn) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**

6.714.1 Detailed Description

Money format ordering data.

This class contains an ordered array of 4 fields to represent the pattern for formatting a money amount. Each field may contain one entry from the part enum. symbol, sign, and value must be present and the remaining field must contain either none or space.

See also

`moneypunct::pos_format()` and `moneypunct::neg_format()` for details of how these fields are interpreted.

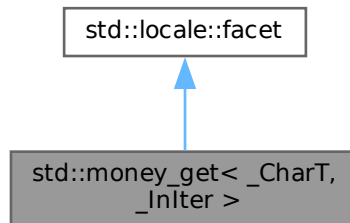
The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

6.715 std::money_get< _CharT, _Inlter > Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::money_get< _CharT, _Inlter >:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Inlter` `iter_type`
- typedef `basic_string< _CharT >` `string_type`

Public Member Functions

- `money_get` (`size_t __refs=0`)
- `template<bool _Intl>`
`_GLIBCXX_BEGIN_NAMESPACE_LDBL_OR_CXX11 _Inlter _M_extract` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `string &__units`) `const`
- `iter_type get` (`iter_type __s`, `iter_type __end`, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `long double &__units`) `const`
- `iter_type get` (`iter_type __s`, `iter_type __end`, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `string_type &__digits`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~money_get` ()
- `template<bool _Intl>`
`iter_type _M_extract` (`iter_type __s`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `string &__digits`) `const`
- virtual `iter_type do_get` (`iter_type __s`, `iter_type __end`, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `long double &__units`) `const`
- virtual `iter_type do_get` (`iter_type __s`, `iter_type __end`, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `string_type &__digits`) `const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_type_c_locale (__c_locale __cloc, const char *__s)`

6.715.1 Detailed Description

```
template<typename _CharT, typename _InIter>
class std::money_get< _CharT, _InIter >
```

Primary class template `money_get`.

This facet encapsulates the code to parse and return a monetary amount from a string.

The `money_get` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `money_get` facet.

6.715.2 Member Typedef Documentation

`char_type`

```
template<typename _CharT , typename _InIter >
typedef _CharT std::money_get< _CharT, _InIter >::char_type
Public typedefs.
```

`iter_type`

```
template<typename _CharT , typename _InIter >
typedef _InIter std::money_get< _CharT, _InIter >::iter_type
Public typedefs.
```

`string_type`

```
template<typename _CharT , typename _InIter >
typedef basic_string<_CharT> std::money_get< _CharT, _InIter >::string_type
Public typedefs.
```

6.715.3 Constructor & Destructor Documentation

`money_get()`

```
template<typename _CharT , typename _InIter >
std::money_get< _CharT, _InIter >::money_get (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~money_get()

```
template<typename _CharT , typename _InIter >
virtual std::money_get< _CharT, _InIter >::~~money_get ( ) [inline], [protected], [virtual]
Destructor.
```

6.715.4 Member Function Documentation**do_get()** [1/2]

```
template<typename _CharT , typename _InIter >
_InIter std::money_get< _CharT, _InIter >::do_get (
    iter_type __s,
    iter_type __end,
    bool __intl,
    ios_base & __io,
    ios_base::iostate & __err,
    long double & __units ) const [protected], [virtual]
```

Read and parse a monetary value.

This function reads and parses characters representing a monetary value. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for details.

References [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#).

Referenced by [std::money_get<_CharT, _InIter>::get\(\)](#), and [std::money_get<_CharT, _InIter>::get\(\)](#).

do_get() [2/2]

```
template<typename _CharT , typename _InIter >
_InIter std::money_get< _CharT, _InIter >::do_get (
    iter_type __s,
    iter_type __end,
    bool __intl,
    ios_base & __io,
    ios_base::iostate & __err,
    string_type & __digits ) const [protected], [virtual]
```

Read and parse a monetary value.

This function reads and parses characters representing a monetary value. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for details.

References [std::ios_base::M_getloc\(\)](#), [std::basic_string<_CharT, _Traits, _Alloc>::resize\(\)](#), and [std::__ctype_abstract_base<_CharT](#)

get() [1/2]

```
template<typename _CharT , typename _InIter >
iter_type std::money_get< _CharT, _InIter >::get (
    iter_type __s,
    iter_type __end,
    bool __intl,
    ios_base & __io,
```

```
ios_base::iostate & __err,
long double & __units ) const [inline]
```

Read and parse a monetary value.

This function reads characters from `__s`, interprets them as a monetary value according to `money_punct` and `ctype` facets retrieved from `io.getloc()`, and returns the result in `units` as an integral value `money_punct::frac_digits()` * the actual amount. For example, the string \$10.01 in a US locale would store 1001 in `units`.

Any characters not part of a valid money amount are not consumed.

If a money value cannot be parsed from the input stream, sets `err=(err|io.failbit)`. If the stream is consumed before finishing parsing, sets `err=(err|io.failbit|io.eofbit)`. `units` is unchanged if parsing fails.

This function works by returning the result of `do_get()`.

Parameters

<code>__s</code>	Start of characters to parse.
<code>__end</code>	End of characters to parse.
<code>__intl</code>	Parameter to use <code>_facet<money_punct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__err</code>	Error field to set if parsing fails.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator referencing first character beyond valid money amount.

References `std::money_get<_CharT, _InIter>::do_get()`.

get() [2/2]

```
template<typename _CharT , typename _InIter >
iter_type std::money_get< _CharT, _InIter >::get (
    iter_type __s,
    iter_type __end,
    bool __intl,
    ios_base & __io,
    ios_base::iostate & __err,
    string_type & __digits ) const [inline]
```

Read and parse a monetary value.

This function reads characters from `__s`, interprets them as a monetary value according to `money_punct` and `ctype` facets retrieved from `io.getloc()`, and returns the result in `digits`. For example, the string \$10.01 in a US locale would store 1001 in `digits`.

Any characters not part of a valid money amount are not consumed.

If a money value cannot be parsed from the input stream, sets `err=(err|io.failbit)`. If the stream is consumed before finishing parsing, sets `err=(err|io.failbit|io.eofbit)`.

This function works by returning the result of `do_get()`.

Parameters

<code>__s</code>	Start of characters to parse.
<code>__end</code>	End of characters to parse.
<code>__intl</code>	Parameter to use <code>_facet<money_punct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__err</code>	Error field to set if parsing fails.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator referencing first character beyond valid money amount.

References `std::money_get<_CharT, _InIter>::do_get()`.

6.715.5 Member Data Documentation

id

```
template<typename _CharT, typename _InIter>
locale::id std::money_get<_CharT, _InIter>::id [static]
Numpunct facet id.
```

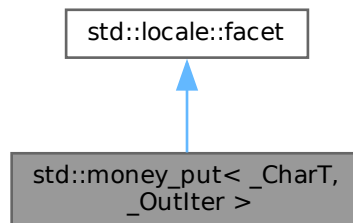
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

6.716 `std::money_put<_CharT, _OutIter>` Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::money_put<_CharT, _OutIter>`:



Public Types

- typedef `_CharT` [char_type](#)
- typedef `_OutIter` [iter_type](#)
- typedef [basic_string<_CharT>](#) [string_type](#)

Public Member Functions

- [money_put](#) (size_t __refs=0)
- template<bool __intl>
`_OutIter` [M_insert](#) ([iter_type](#) __s, [ios_base](#) &__io, [char_type](#) __fill, const [string_type](#) &__digits) const
- [iter_type](#) put ([iter_type](#) __s, bool __intl, [ios_base](#) &__io, [char_type](#) __fill, const [string_type](#) &__digits) const
- [iter_type](#) put ([iter_type](#) __s, bool __intl, [ios_base](#) &__io, [char_type](#) __fill, long double __units) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual `~money_put()`
- `template<bool _Intl>`
`iter_type _M_insert(iter_type __s, ios_base & __io, char_type __fill, const string_type & __digits) const`
- virtual `iter_type do_put(iter_type __s, bool __intl, ios_base & __io, char_type __fill, const string_type & __digits) const`
- virtual `iter_type do_put(iter_type __s, bool __intl, ios_base & __io, char_type __fill, long double __units) const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale & __cloc) throw()`
- static void `_S_create_c_locale(__c_locale & __cloc, const char * __s, __c_locale __old=0)`
- static void `_S_destroy_c_locale(__c_locale & __cloc)`
- static `__c_locale _S_get_c_locale()`
- static const char * `_S_get_c_name()` throw()
- static `__c_locale _S_lc_type_c_locale(__c_locale __cloc, const char * __s)`

6.716.1 Detailed Description

```
template<typename _CharT, typename _OutIter>
class std::money_put< _CharT, _OutIter >
```

Primary class template `money_put`.

This facet encapsulates the code to format and output a monetary amount.

The `money_put` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `money_put` facet.

6.716.2 Member Typedef Documentation

`char_type`

```
template<typename _CharT , typename _OutIter >
typedef _CharT std::money_put< _CharT, _OutIter >::char_type
```

Public typedefs.

`iter_type`

```
template<typename _CharT , typename _OutIter >
typedef _OutIter std::money_put< _CharT, _OutIter >::iter_type
```

Public typedefs.

`string_type`

```
template<typename _CharT , typename _OutIter >
typedef basic_string<_CharT> std::money_put< _CharT, _OutIter >::string_type
```

Public typedefs.

6.716.3 Constructor & Destructor Documentation

`money_put()`

```
template<typename _CharT , typename _OutIter >
std::money_put< _CharT, _OutIter >::money_put (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.
This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~money_put()

```
template<typename _CharT , typename _OutIter >
virtual std::money_put< _CharT, _OutIter >::~~money_put ( ) [inline], [protected], [virtual]
Destructor.
```

6.716.4 Member Function Documentation

do_put() [1/2]

```
template<typename _CharT , typename _OutIter >
_OutIter std::money_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    bool __intl,
    ios_base & __io,
    char_type __fill,
    const string_type & __digits ) const [protected], [virtual]
```

Format and output a monetary value.

This function formats *digits* as a monetary value according to `money_punct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the string `1001` in a US locale would write `$10.01` to `__s`. This function is a hook for derived classes to change the value returned.

See also

`put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<money_punct<CharT,intl>> .</code>
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator after writing.

do_put() [2/2]

```
template<typename _CharT , typename _OutIter >
_OutIter std::money_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    bool __intl,
    ios_base & __io,
```



```
char_type __fill,
long double __units ) const [protected], [virtual]
```

Format and output a monetary value.

This function formats *units* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the value 1001 in a US locale would write \$10.01 to `__s`.

This function is a hook for derived classes to change the value returned.

See also

`put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl> ></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator after writing.

References `std::ios_base::getloc()`, and `std::__ctype_abstract_base<_CharT >::widen()`.

Referenced by `std::money_put<_CharT, _OutIter >::put()`, and `std::money_put<_CharT, _OutIter >::put()`.

`put()` [1/2]

```
template<typename _CharT , typename _OutIter >
iter_type std::money_put< _CharT, _OutIter >::put (
    iter_type __s,
    bool __intl,
    ios_base & __io,
    char_type __fill,
    const string_type & __digits ) const [inline]
```

Format and output a monetary value.

This function formats *digits* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the string 1001 in a US locale would write \$10.01 to `__s`.

This function works by returning the result of `do_put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl> ></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator after writing.

References `std::money_put<_CharT, _OutIter >::do_put()`.

put() [2/2]

```
template<typename _CharT , typename _OutIter >
iter_type std::money_put< _CharT, _OutIter >::put (
    iter_type __s,
    bool __intl,
    ios_base & __io,
    char_type __fill,
    long double __units ) const [inline]
```

Format and output a monetary value.

This function formats *units* as a monetary value according to moneypunct and ctype facets retrieved from io.getloc(), and writes the resulting characters to __s. For example, the value 1001 in a US locale would write \$10.01 to __s.

This function works by returning the result of do_put().

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator after writing.

References [std::money_put<_CharT, _OutIter>::do_put\(\)](#).

6.716.5 Member Data Documentation**id**

```
template<typename _CharT , typename _OutIter >
locale::id std::money_put< _CharT, _OutIter >::id [static]
```

Numpunct facet id.

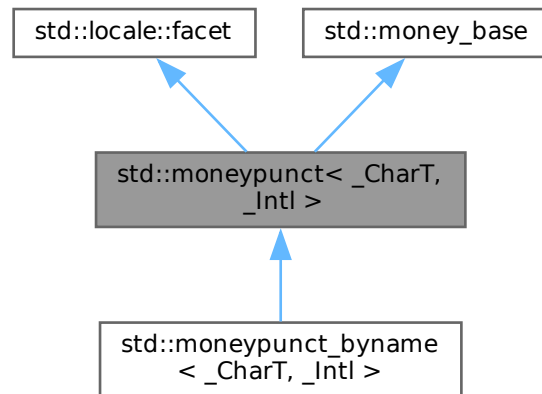
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

6.717 std::moneypunct<_CharT, _Intl> Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::moneypunct< _CharT, _Intl >`:



Public Types

- enum { **`_S_minus`** , **`_S_zero`** , **`_S_end`** }
- typedef `__moneypunct_cache< _CharT, _Intl > __cache_type`
- enum **`part`** {
 `none` , **`space`** , **`symbol`** , **`sign`** ,
 `value` }
- typedef `_CharT char_type`
- typedef `basic_string< _CharT > string_type`

Public Member Functions

- `moneypunct` (`__c_locale __cloc`, `const char * __s`, `size_t __refs=0`)
- `moneypunct` (`__cache_type * __cache`, `size_t __refs=0`)
- `moneypunct` (`size_t __refs=0`)
- `string_type curr_symbol` () const
- `char_type decimal_point` () const
- `int frac_digits` () const
- `string grouping` () const
- `string_type negative_sign` () const
- `string_type positive_sign` () const
- `char_type thousands_sep` () const
- pattern `pos_format` () const
- pattern `neg_format` () const

Static Public Member Functions

- static pattern **`_S_construct_pattern`** (`char __precedes`, `char __space`, `char __posn`) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**
- static [locale::id](#) **id**
- static const bool **intl**

Protected Member Functions

- virtual [~moneypunct](#) ()
- void **_M_initialize_moneypunct** (__c_locale __cloc=0, const char *__name=0)
- void **_M_initialize_moneypunct** (__c_locale, const char *)
- void **_M_initialize_moneypunct** (__c_locale, const char *)
- void **_M_initialize_moneypunct** (__c_locale, const char *)
- void **_M_initialize_moneypunct** (__c_locale, const char *)
- virtual [string_type](#) **do_curr_symbol** () const
- virtual [char_type](#) **do_decimal_point** () const
- virtual int **do_frac_digits** () const
- virtual [string](#) **do_grouping** () const
- virtual pattern **do_neg_format** () const
- virtual [string_type](#) **do_negative_sign** () const
- virtual pattern **do_pos_format** () const
- virtual [string_type](#) **do_positive_sign** () const
- virtual [char_type](#) **do_thousands_sep** () const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

6.717.1 Detailed Description

```
template<typename _CharT, bool _Intl>
class std::moneypunct<_CharT, _Intl>
```

Primary class template moneypunct.

This facet encapsulates the punctuation, grouping and other formatting features of money amount string representations.

6.717.2 Member Typedef Documentation

char_type

```
template<typename _CharT, bool _Intl>
typedef _CharT std::moneypunct<_CharT, _Intl>::char_type
```

Public typedefs.

string_type

```
template<typename _CharT, bool _Intl>
typedef basic\_string<_CharT> std::moneypunct<_CharT, _Intl>::string_type
```

Public typedefs.

6.717.3 Constructor & Destructor Documentation

moneypunct() [1/3]

```
template<typename _CharT , bool _Intl>
std::moneypunct< _CharT, _Intl >::moneypunct (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

moneypunct() [2/3]

```
template<typename _CharT , bool _Intl>
std::moneypunct< _CharT, _Intl >::moneypunct (
    __cache_type * __cache,
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is an internal constructor.

Parameters

<code>__cache</code>	Cache for optimization.
<code>__refs</code>	Passed to the base facet class.

moneypunct() [3/3]

```
template<typename _CharT , bool _Intl>
std::moneypunct< _CharT, _Intl >::moneypunct (
    __c_locale __cloc,
    const char * __s,
    size_t __refs = 0 ) [inline], [explicit]
```

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__s</code>	The name of a locale.
<code>__refs</code>	Passed to the base facet class.

~moneypunct()

```
template<typename _CharT , bool _Intl>
virtual std::moneypunct< _CharT, _Intl >::~~moneypunct ( ) [protected], [virtual]
```

Destructor.

6.717.4 Member Function Documentation

curr_symbol()

```
template<typename _CharT , bool _Intl>
string_type std::moneypunct< _CharT, _Intl >::curr_symbol ( ) const [inline]
```

Return currency symbol string.

This function returns a string_type to use as a currency symbol. It does so by returning returning moneypunct<char↵_type>::do_curr_symbol().

Returns

string_type representing a currency symbol.

References [std::moneypunct< _CharT, _Intl >::do_curr_symbol\(\)](#).

decimal_point()

```
template<typename _CharT , bool _Intl>
char_type std::moneypunct< _CharT, _Intl >::decimal_point ( ) const [inline]
```

Return decimal point character.

This function returns a char_type to use as a decimal point. It does so by returning returning moneypunct<char↵_type>::do_decimal_point().

Returns

char_type representing a decimal point.

References [std::moneypunct< _CharT, _Intl >::do_decimal_point\(\)](#).

do_curr_symbol()

```
template<typename _CharT , bool _Intl>
virtual string_type std::moneypunct< _CharT, _Intl >::do_curr_symbol ( ) const [inline], [protected], [virtual]
```

Return currency symbol string.

This function returns a string_type to use as a currency symbol. This function is a hook for derived classes to change the value returned.

See also

[curr_symbol\(\)](#) for details.

Returns

string_type representing a currency symbol.

Referenced by [std::moneypunct< _CharT, _Intl >::curr_symbol\(\)](#).

do_decimal_point()

```
template<typename _CharT , bool _Intl>
virtual char_type std::moneypunct< _CharT, _Intl >::do_decimal_point ( ) const [inline], [protected], [virtual]
```

Return decimal point character.

Returns a char_type to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Referenced by [std::moneypunct< _CharT, _Intl >::decimal_point\(\)](#).

do_frac_digits()

```
template<typename _CharT , bool _Intl>
virtual int std::moneypunct< _CharT, _Intl >::do_frac_digits ( ) const [inline], [protected],
[virtual]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. This function is a hook for derived classes to change the value returned.

See also

[frac_digits\(\)](#) for details.

Returns

Number of digits in amount fraction.

Referenced by [std::moneypunct< _CharT, _Intl >::frac_digits\(\)](#).

do_grouping()

```
template<typename _CharT , bool _Intl>
virtual string std::moneypunct< _CharT, _Intl >::do_grouping ( ) const [inline], [protected],
[virtual]
```

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

[grouping\(\)](#) for details.

Returns

String representing grouping specification.

Referenced by [std::moneypunct< _CharT, _Intl >::grouping\(\)](#).

do_neg_format()

```
template<typename _CharT , bool _Intl>
virtual pattern std::moneypunct< _CharT, _Intl >::do_neg_format ( ) const [inline], [protected],
[virtual]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a negative valued money amount. This function is a hook for derived classes to change the value returned.

See also

[neg_format\(\)](#) for details.

Returns

Pattern for money values.

Referenced by [std::moneypunct< _CharT, _Intl >::neg_format\(\)](#).

do_negative_sign()

```
template<typename _CharT , bool _Intl>
virtual string\_type std::moneypunct< _CharT, _Intl >::do_negative_sign ( ) const [inline], [protected],
[virtual]
```

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. This function is a hook for derived classes to change the value returned.

See also

`negative_sign()` for details.

Returns

string_type representing a negative sign.

Referenced by [std::moneypunct<_CharT, _Intl>::negative_sign\(\)](#).

do_pos_format()

```
template<typename _CharT , bool _Intl>
virtual pattern std::moneypunct< _CharT, _Intl >::do_pos_format ( ) const [inline], [protected],
[virtual]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive valued money amount. This function is a hook for derived classes to change the value returned.

See also

`pos_format()` for details.

Returns

Pattern for money values.

Referenced by [std::moneypunct<_CharT, _Intl>::pos_format\(\)](#).

do_positive_sign()

```
template<typename _CharT , bool _Intl>
virtual string\_type std::moneypunct< _CharT, _Intl >::do_positive_sign ( ) const [inline], [protected],
[virtual]
```

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. This function is a hook for derived classes to change the value returned.

See also

`positive_sign()` for details.

Returns

string_type representing a positive sign.

Referenced by [std::moneypunct<_CharT, _Intl>::positive_sign\(\)](#).

do_thousands_sep()

```
template<typename _CharT , bool _Intl>
virtual char_type std::moneypunct< _CharT, _Intl >::do_thousands_sep ( ) const [inline], [protected],
[virtual]
```

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Referenced by `std::moneypunct< _CharT, _Intl >::thousands_sep()`.

frac_digits()

```
template<typename _CharT , bool _Intl>
int std::moneypunct< _CharT, _Intl >::frac_digits ( ) const [inline]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. It does so by returning `std::moneypunct<char_type>::do_frac_digits()`.

The fractional part of a money amount is optional. But if it is present, there must be `frac_digits()` digits.

Returns

Number of digits in amount fraction.

References `std::moneypunct< _CharT, _Intl >::do_frac_digits()`.

grouping()

```
template<typename _CharT , bool _Intl>
string std::moneypunct< _CharT, _Intl >::grouping ( ) const [inline]
```

Return grouping specification.

This function returns a string representing groupings for the integer part of an amount. Groupings indicate where thousands separators should be inserted.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the `grouping()` returns `\003\002` and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was 32, this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling `std::moneypunct<char_type>::do_grouping()`.

Returns

string representing grouping specification.

References `std::moneypunct< _CharT, _Intl >::do_grouping()`.

neg_format()

```
template<typename _CharT , bool _Intl>
pattern std::moneypunct< _CharT, _Intl >::neg_format ( ) const [inline]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `std::moneypunct<char_type>::do_pos_format()` or `std::moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field. The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

References [std::moneypunct<_CharT, _Intl>::do_neg_format\(\)](#).

negative_sign()

```
template<typename _CharT, bool _Intl>
string_type std::moneypunct<_CharT, _Intl>::negative_sign ( ) const [inline]
```

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. It does so by returning `moneypunct<char_type>::do_negative_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `neg_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a negative sign.

References [std::moneypunct<_CharT, _Intl>::do_negative_sign\(\)](#).

pos_format()

```
template<typename _CharT, bool _Intl>
pattern std::moneypunct<_CharT, _Intl>::pos_format ( ) const [inline]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

References [std::moneypunct<_CharT, _Intl>::do_pos_format\(\)](#).

positive_sign()

```
template<typename _CharT, bool _Intl>
string_type std::moneypunct<_CharT, _Intl>::positive_sign ( ) const [inline]
```

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. It does so by returning `money_punct<char_type>::do_positive_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `pos_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a positive sign.

References [std::money_punct<_CharT, _Intl>::do_positive_sign\(\)](#).

thousands_sep()

```
template<typename _CharT, bool _Intl>
char_type std::money_punct<_CharT, _Intl>::thousands_sep ( ) const [inline]
```

Return thousands separator character.

This function returns a `char_type` to use as a thousands separator. It does so by returning `money_punct<char_type>::do_thousands_sep()`.

Returns

`char_type` representing a thousands separator.

References [std::money_punct<_CharT, _Intl>::do_thousands_sep\(\)](#).

6.717.5 Member Data Documentation

id

```
template<typename _CharT, bool _Intl>
locale::id std::money_punct<_CharT, _Intl>::id [static]
Numpunct facet id.
```

intl

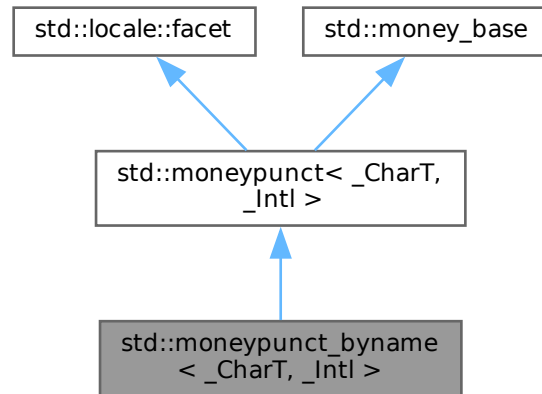
```
template<typename _CharT, bool _Intl>
const bool std::money_punct<_CharT, _Intl>::intl [static]
This value is provided by the standard, but no reason for its existence.
The documentation for this class was generated from the following file:
```

- [locale_facets_nonio.h](#)

6.718 std::money_punct_byname<_CharT, _Intl> Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::moneypunct_byname<_CharT, _Intl>:



Public Types

- enum { **_S_minus** , **_S_zero** , **_S_end** }
- typedef __moneypunct_cache<_CharT, _Intl> **__cache_type**
- typedef _CharT **char_type**
- enum **part** {
 none , **space** , **symbol** , **sign** ,
 value }
- typedef [basic_string](#)<_CharT> **string_type**

Public Member Functions

- **moneypunct_byname** (const char *__s, size_t __refs=0)
- **moneypunct_byname** (const [string](#) &__s, size_t __refs=0)
- [string_type](#) **curr_symbol** () const
- char_type **decimal_point** () const
- int **frac_digits** () const
- [string](#) **grouping** () const
- [string_type](#) **negative_sign** () const
- [string_type](#) **positive_sign** () const
- char_type **thousands_sep** () const
- pattern [pos_format](#) () const
- pattern [neg_format](#) () const

Static Public Member Functions

- static pattern **_S_construct_pattern** (char __precedes, char __space, char __posn) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**
- static [locale::id](#) **id**
- static const bool **intl**

Protected Member Functions

- void **_M_initialize_moneypunct** ([__c_locale](#) __cloc=0, const char *__name=0)
- void **_M_initialize_moneypunct** ([__c_locale](#), const char *)
- void **_M_initialize_moneypunct** ([__c_locale](#), const char *)
- void **_M_initialize_moneypunct** ([__c_locale](#), const char *)
- void **_M_initialize_moneypunct** ([__c_locale](#), const char *)
- virtual [string_type](#) **do_curr_symbol** () const
- virtual char_type **do_decimal_point** () const
- virtual int **do_frac_digits** () const
- virtual [string](#) **do_grouping** () const
- virtual pattern **do_neg_format** () const
- virtual [string_type](#) **do_negative_sign** () const
- virtual pattern **do_pos_format** () const
- virtual [string_type](#) **do_positive_sign** () const
- virtual char_type **do_thousands_sep** () const

Static Protected Member Functions

- static [__c_locale](#) **_S_clone_c_locale** ([__c_locale](#) &__cloc) throw ()
- static void **_S_create_c_locale** ([__c_locale](#) &__cloc, const char *__s, [__c_locale](#) __old=0)
- static void **_S_destroy_c_locale** ([__c_locale](#) &__cloc)
- static [__c_locale](#) **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static [__c_locale](#) **_S_lc_ctype_c_locale** ([__c_locale](#) __cloc, const char *__s)

6.718.1 Detailed Description

```
template<typename _CharT, bool _Intl>
class std::moneypunct_byname< _CharT, _Intl >
```

class moneypunct_byname [22.2.6.4].

6.718.2 Member Function Documentation

curr_symbol()

```
template<typename _CharT , bool _Intl>
string\_type std::moneypunct< _CharT, _Intl >::curr_symbol ( ) const [inline], [inherited]
```

Return currency symbol string.

This function returns a [string_type](#) to use as a currency symbol. It does so by returning returning moneypunct<char←
_type>::do_curr_symbol().

Returns

string_type representing a currency symbol.

References [std::moneypunct< _CharT, _Intl >::do_curr_symbol\(\)](#).

decimal_point()

```
template<typename _CharT, bool _Intl>
char_type std::moneypunct<_CharT, _Intl>::decimal_point ( ) const [inline], [inherited]
```

Return decimal point character.

This function returns a `char_type` to use as a decimal point. It does so by returning `returning moneypunct<char_↵type>::do_decimal_point()`.

Returns

char_type representing a decimal point.

References `std::moneypunct<_CharT, _Intl>::do_decimal_point()`.

do_curr_symbol()

```
template<typename _CharT, bool _Intl>
virtual string_type std::moneypunct<_CharT, _Intl>::do_curr_symbol ( ) const [inline], [protected],
[virtual], [inherited]
```

Return currency symbol string.

This function returns a `string_type` to use as a currency symbol. This function is a hook for derived classes to change the value returned.

See also

`curr_symbol()` for details.

Returns

string_type representing a currency symbol.

Referenced by `std::moneypunct<_CharT, _Intl>::curr_symbol()`.

do_decimal_point()

```
template<typename _CharT, bool _Intl>
virtual char_type std::moneypunct<_CharT, _Intl>::do_decimal_point ( ) const [inline], [protected],
[virtual], [inherited]
```

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Referenced by `std::moneypunct<_CharT, _Intl>::decimal_point()`.

do_frac_digits()

```
template<typename _CharT, bool _Intl>
virtual int std::moneypunct<_CharT, _Intl>::do_frac_digits ( ) const [inline], [protected],
[virtual], [inherited]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. This function is a hook for derived classes to change the value returned.

See also

`frac_digits()` for details.

Returns

Number of digits in amount fraction.

Referenced by [std::moneypunct<_CharT, _Intl>::frac_digits\(\)](#).

do_grouping()

```
template<typename _CharT, bool _Intl>
virtual string std::moneypunct<_CharT, _Intl>::do_grouping ( ) const [inline], [protected],
[virtual], [inherited]
```

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

[grouping\(\)](#) for details.

Returns

String representing grouping specification.

Referenced by [std::moneypunct<_CharT, _Intl>::grouping\(\)](#).

do_neg_format()

```
template<typename _CharT, bool _Intl>
virtual pattern std::moneypunct<_CharT, _Intl>::do_neg_format ( ) const [inline], [protected],
[virtual], [inherited]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a negative valued money amount. This function is a hook for derived classes to change the value returned.

See also

[neg_format\(\)](#) for details.

Returns

Pattern for money values.

Referenced by [std::moneypunct<_CharT, _Intl>::neg_format\(\)](#).

do_negative_sign()

```
template<typename _CharT, bool _Intl>
virtual string_type std::moneypunct<_CharT, _Intl>::do_negative_sign ( ) const [inline], [protected],
[virtual], [inherited]
```

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. This function is a hook for derived classes to change the value returned.

See also

[negative_sign\(\)](#) for details.

Returns

string_type representing a negative sign.

Referenced by [std::moneypunct<_CharT, _Intl>::negative_sign\(\)](#).

do_pos_format()

```
template<typename _CharT, bool _Intl>
virtual pattern std::moneypunct<_CharT, _Intl>::do_pos_format ( ) const [inline], [protected],
[virtual], [inherited]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive valued money amount. This function is a hook for derived classes to change the value returned.

See also

`pos_format()` for details.

Returns

Pattern for money values.

Referenced by `std::moneypunct<_CharT, _Intl>::pos_format()`.

do_positive_sign()

```
template<typename _CharT, bool _Intl>
virtual string_type std::moneypunct<_CharT, _Intl>::do_positive_sign ( ) const [inline], [protected],
[virtual], [inherited]
```

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. This function is a hook for derived classes to change the value returned.

See also

`positive_sign()` for details.

Returns

string_type representing a positive sign.

Referenced by `std::moneypunct<_CharT, _Intl>::positive_sign()`.

do_thousands_sep()

```
template<typename _CharT, bool _Intl>
virtual char_type std::moneypunct<_CharT, _Intl>::do_thousands_sep ( ) const [inline], [protected],
[virtual], [inherited]
```

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Referenced by `std::moneypunct<_CharT, _Intl>::thousands_sep()`.

frac_digits()

```
template<typename _CharT, bool _Intl>
int std::moneypunct<_CharT, _Intl>::frac_digits ( ) const [inline], [inherited]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. It does so by returning `std::moneypunct<char_type>::do_frac_digits()`.

The fractional part of a money amount is optional. But if it is present, there must be `frac_digits()` digits.

Returns

Number of digits in amount fraction.

References [std::moneypunct<_CharT, _Intl>::do_frac_digits\(\)](#).

grouping()

```
template<typename _CharT, bool _Intl>
string std::moneypunct<_CharT, _Intl>::grouping ( ) const [inline], [inherited]
```

Return grouping specification.

This function returns a string representing groupings for the integer part of an amount. Groupings indicate where thousands separators should be inserted.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the `grouping()` returns `\003\002` and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was 32, this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling `moneypunct<char_type>::do_grouping()`.

Returns

string representing grouping specification.

References [std::moneypunct<_CharT, _Intl>::do_grouping\(\)](#).

neg_format()

```
template<typename _CharT, bool _Intl>
pattern std::moneypunct<_CharT, _Intl>::neg_format ( ) const [inline], [inherited]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

References [std::moneypunct<_CharT, _Intl>::do_neg_format\(\)](#).

negative_sign()

```
template<typename _CharT, bool _Intl>
string_type std::moneypunct<_CharT, _Intl>::negative_sign ( ) const [inline], [inherited]
```

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. It does so by returning `moneypunct<char_type>::do_negative_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `neg_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a negative sign.

References [std::moneypunct<_CharT, _Intl>::do_negative_sign\(\)](#).

pos_format()

```
template<typename _CharT, bool _Intl>
pattern std::moneypunct<_CharT, _Intl>::pos_format ( ) const [inline], [inherited]
```

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

References [std::moneypunct<_CharT, _Intl>::do_pos_format\(\)](#).

positive_sign()

```
template<typename _CharT, bool _Intl>
string_type std::moneypunct<_CharT, _Intl>::positive_sign ( ) const [inline], [inherited]
```

Return positive sign string.

This function returns a *string_type* to use as a sign for positive amounts. It does so by returning `moneypunct<char_type>::do_positive_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `pos_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a positive sign.

References [std::moneypunct<_CharT, _Intl>::do_positive_sign\(\)](#).

thousands_sep()

```
template<typename _CharT, bool _Intl>
char_type std::moneypunct<_CharT, _Intl>::thousands_sep ( ) const [inline], [inherited]
```

Return thousands separator character.

This function returns a *char_type* to use as a thousands separator. It does so by returning `moneypunct<char_type>::do_thousands_sep()`.

Returns

char_type representing a thousands separator.

References [std::moneypunct<_CharT, _Intl>::do_thousands_sep\(\)](#).

6.718.3 Member Data Documentation

id

```
template<typename _CharT, bool _Intl>
locale::id std::moneypunct< _CharT, _Intl >::id [static], [inherited]
Numpunct facet id.
```

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

6.719 std::move_iterator<_Iterator> Class Template Reference

```
#include <std_iterator.h>
Inherits std::__detail::__move_iter_cat<_Iterator>.
```

Public Types

- using **difference_type** = iter_difference_t<_Iterator>
- using **iterator_concept** = decltype(_S_iter_concept())
- using **iterator_type** = _Iterator
- using **pointer** = _Iterator
- using **reference** = iter_rvalue_reference_t<_Iterator>
- using **value_type** = iter_value_t<_Iterator>

Public Member Functions

- template<typename _Iter>
requires __convertible<_Iter>
constexpr **move_iterator** (const [move_iterator](#)<_Iter> &__i)
- constexpr **move_iterator** (iterator_type __i)
- constexpr iterator_type **base** () &&
- constexpr const iterator_type & **base** () const &noexcept
- constexpr reference **operator*** () const
- constexpr [move_iterator](#) **operator+** (difference_type __n) const
- constexpr [move_iterator](#) & **operator++** ()
- constexpr [move_iterator](#) **operator++** (int)
- constexpr void **operator++** (int)
- constexpr [move_iterator](#) & **operator+=** (difference_type __n)
- constexpr [move_iterator](#) **operator-** (difference_type __n) const
- constexpr [move_iterator](#) & **operator--** ()
- constexpr [move_iterator](#) **operator--** (int)
- constexpr [move_iterator](#) & **operator-=** (difference_type __n)
- constexpr pointer **operator->** () const
- template<typename _Iter>
requires __convertible<_Iter> && assignable_from<_Iterator&, const _Iter&>
constexpr [move_iterator](#) & **operator=** (const [move_iterator](#)<_Iter> &__i)
- constexpr reference **operator[]** (difference_type __n) const

Friends

- `constexpr iter_rvalue_reference_t< _Iterator > iter_move` (const [move_iterator](#) &__i) noexcept(noexcept(ranges↵↵::iter_move(__i._M_current)))
- `template<indirectly_swappable< _Iterator > _Iter2>`
`constexpr void iter_swap` (const [move_iterator](#) &__x, const [move_iterator](#)< _Iter2 > &__y) noexcept(noexcept(ranges↵↵::iter_swap(__x._M_current, __y._M_current)))
- `template<sized_sentinel_for< _Iterator > _Sent>`
`constexpr iter_difference_t< _Iterator > operator-` (const [move_iterator](#) &__x, const `move_sentinel`< _Sent > &__y)
- `template<sized_sentinel_for< _Iterator > _Sent>`
`constexpr iter_difference_t< _Iterator > operator-` (const `move_sentinel`< _Sent > &__x, const [move_iterator](#) &__y)
- `template<sentinel_for< _Iterator > _Sent>`
`constexpr bool operator==` (const [move_iterator](#) &__x, const `move_sentinel`< _Sent > &__y)

6.719.1 Detailed Description

```
template<typename _Iterator>
class std::move_iterator< _Iterator >
```

Class template `move_iterator` is an iterator adapter with the same behavior as the underlying iterator except that its dereference operator implicitly converts the value returned by the underlying iterator's dereference operator to an rvalue reference. Some generic algorithms can be called with move iterators to replace copying with moving.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.720 `std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>` Class Template Reference

```
#include <functional>
Inherits _Mofunc_base.
```

Public Types

- using `result_type` = `_Res`

Public Member Functions

- [move_only_function](#) () noexcept
- `template<typename _Fn , typename _Vt = decay_t<_Fn>>`
requires (!is_same_v<_Vt, move_only_function>) && (!is_in_place_type_v<_Vt>) && __is_callable_from<_Vt>
[move_only_function](#) (_Fn &&__f) noexcept(_S_nothrow_init< _Vt, _Fn >())
- `template<typename _Tp , typename... _Args>`
requires is_constructible_v<_Tp, _Args...> && __is_callable_from<_Tp>
[move_only_function](#) (in_place_type_t< _Tp >, _Args &&... __args) noexcept(_S_nothrow_init< _Tp, _Args... >())
- `template<typename _Tp , typename _Up , typename... _Args>`
requires is_constructible_v<_Tp, initializer_list<_Up>&, _Args...> && __is_callable_from<_Tp>
[move_only_function](#) (in_place_type_t< _Tp >, initializer_list< _Up > __il, _Args &&... __args) noexcept(_S↵↵nothrow_init< _Tp, initializer_list< _Up > &, _Args... >())
- [move_only_function](#) (move_only_function &&__x) noexcept
- [move_only_function](#) (nullptr_t) noexcept
- [operator bool](#) () const noexcept

- `_Res operator() (_ArgTypes... __args) _GLIBCXX_MOF_CV noexcept(_Noex)`
- `template<typename _Fn >`
`requires is_constructible_v<move_only_function, _Fn>`
`move_only_function & operator= (_Fn &&__f) noexcept(is_nothrow_constructible_v< move_only_function, _Fn >)`
- `move_only_function & operator= (move_only_function &&__x) noexcept`
- `move_only_function & operator= (nullptr_t) noexcept`
- `void swap (move_only_function &__x) noexcept`

Friends

- `bool operator== (const move_only_function &__x, nullptr_t) noexcept`
- `void swap (move_only_function &__x, move_only_function &__y) noexcept`

6.720.1 Detailed Description

`template<typename _Res, typename... _ArgTypes, bool _Noex>`
class `std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>`

Polymorphic function wrapper.

Since

C++23

The `std::move_only_function` class template is a call wrapper similar to `std::function`, but does not require the stored target function to be copyable.

It also supports const-qualification, ref-qualification, and no-throw guarantees. The qualifications and exception-specification of the `move_only_function::operator()` member function are respected when invoking the target function.

6.720.2 Constructor & Destructor Documentation

`move_only_function()` [1/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
( ) [inline], [noexcept]
```

Creates an empty object.

`move_only_function()` [2/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
(
    nullptr_t ) [inline], [noexcept]
```

Creates an empty object.

`move_only_function()` [3/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
(
    move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> && __x )
[inline], [noexcept]
```

Moves the target object, leaving the source empty.

`move_only_function()` [4/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
template<typename _Fn , typename _Vt = decay_t<_Fn>>
requires (!is_same_v<_Vt, move_only_function>) && (!__is_in_place_type_v<_Vt>) && __is_callable<
_from<_Vt>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
(
    _Fn && __f ) [inline], [noexcept]
```

Stores a target object initialized from the argument.

`move_only_function()` [5/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
template<typename _Tp , typename... _Args>
requires is_constructible_v<_Tp, _Args...> && __is_callable_from<_Tp>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
(
    in_place_type_t< _Tp > ,
    _Args &&... __args ) [inline], [explicit], [noexcept]
```

Stores a target object initialized from the arguments.

`move_only_function()` [6/6]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
template<typename _Tp , typename _Up , typename... _Args>
requires is_constructible_v<_Tp, initializer_list<_Up>&, _Args...> && __is_callable_from<_Tp>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::move_only_function
(
    in_place_type_t< _Tp > ,
    initializer_list< _Up > __il,
    _Args &&... __args ) [inline], [explicit], [noexcept]
```

Stores a target object initialized from the arguments.

6.720.3 Member Function Documentation

`operator bool()`

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::operator bool ( )
const [inline], [explicit], [noexcept]
```

True if a target object is present, false otherwise.

`operator()()`

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
_Res std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::operator() (
    _ArgTypes... __args ) [inline], [noexcept]
```

Invoke the target object.

The target object will be invoked using the supplied arguments, and as an lvalue or rvalue, and as const or non-const, as dictated by the template arguments of the `move_only_function` specialization.

Precondition

Must not be empty.

operator=() [1/3]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
template<typename _Fn >
requires is_constructible_v<move_only_function, _Fn>
move_only_function & std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_↵
Noex)>::operator= (
    _Fn && __f ) [inline], [noexcept]
```

Stores a new target object, initialized from the argument.

operator=() [2/3]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
move_only_function & std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_↵
Noex)>::operator= (
    move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> && __x )
[inline], [noexcept]
```

Stores a new target object, leaving x empty.

operator=() [3/3]

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
move_only_function & std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_↵
Noex)>::operator= (
    nullptr_t ) [inline], [noexcept]
```

Destroys the target object (if any).

swap()

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
void std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>::swap (
    move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> & __x )
[inline], [noexcept]
```

Exchange the target objects (if any).

References [std::swap\(\)](#).

6.720.4 Friends And Related Symbol Documentation**operator==**

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
bool operator== (
    const move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> & _↵
_x,
    nullptr_t ) [friend]
```

Check for emptiness by comparing with nullptr.

swap

```
template<typename _Res , typename... _ArgTypes, bool _Noex>
void swap (
    move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> & __x,
    move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)> & __y )
[friend]
```

Exchange the target objects (if any).

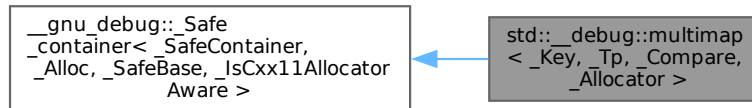
The documentation for this class was generated from the following file:

- [mofunc_impl.h](#)

6.721 std::__debug::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference

```
#include <multimap.h>
```

Inheritance diagram for std::__debug::multimap< _Key, _Tp, _Compare, _Allocator >:



Public Types

- typedef _Allocator **allocator_type**
- typedef [__gnu_debug::__Safe_iterator](#)< [_Base_const_iterator](#), [multimap](#) > **const_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< [const_iterator](#) > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef [__gnu_debug::__Safe_iterator](#)< [_Base_iterator](#), [multimap](#) > **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- typedef _Tp **mapped_type**
- using **node_type** = typename _Base::node_type
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< [iterator](#) > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef [std::pair](#)< const _Key, _Tp > **value_type**

Public Member Functions

- **multimap** (_Base_ref __x)
- template<typename _InputIterator >
multimap (_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator())
- template<typename _InputIterator >
multimap (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- **multimap** (const _Compare &__comp, const _Allocator &__a=_Allocator())
- **multimap** (const allocator_type &__a)
- **multimap** (const [multimap](#) &)=default
- **multimap** (const [multimap](#) &__m, const __type_identity_t< allocator_type > &__a)
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const _Compare &__c=_Compare(), const allocator_type &__a=allocator_type())
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const allocator_type &__a)

- **multimap** ([multimap](#) &&)=default
- **multimap** ([multimap](#) &&__m, const __type_identity_t< allocator_type > &__a) noexcept(noexcept([_Base](#)(std::move(↵__m), __a)))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) **begin** () const noexcept
- [iterator](#) **begin** () noexcept
- [const_iterator](#) **cbegin** () const noexcept
- [const_iterator](#) **cend** () const noexcept
- void **clear** () noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- template<typename... _Args>
[iterator](#) **emplace** (_Args &&... __args)
- template<typename... _Args>
[iterator](#) **emplace_hint** (const [iterator](#) __pos, _Args &&... __args)
- [const_iterator](#) **end** () const noexcept
- [iterator](#) **end** () noexcept
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const _Kt &__x) const
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const key_type &__x)
- [std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const key_type &__x) const
- [_Base_iterator](#) **erase** ([_Base_const_iterator](#) __position)
- size_type **erase** (const key_type &__x)
- [iterator](#) **erase** (const [iterator](#) __first, [const_iterator](#) __last)
- [iterator](#) **erase** (const [iterator](#) __position)
- _GLIBCXX_ABI_TAG_CXX11 [iterator](#) **erase** ([iterator](#) __position)
- node_type **extract** (const key_type &__key)
- node_type **extract** ([const_iterator](#) __position)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[iterator](#) **find** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[const_iterator](#) **find** (const _Kt &__x) const
- [iterator](#) **find** (const key_type &__x)
- [const_iterator](#) **find** (const key_type &__x) const
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[iterator](#) **insert** (_Pair &&__x)
- [iterator](#) **insert** (const value_type &__x)
- [iterator](#) **insert** (const [iterator](#) __hint, node_type &&__nh)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[iterator](#) **insert** (const [iterator](#) __position, _Pair &&__x)
- [iterator](#) **insert** (const [iterator](#) __position, const value_type &__x)
- [iterator](#) **insert** (const [iterator](#) __position, value_type &&__x)
- [iterator](#) **insert** (node_type &&__nh)
- void **insert** (std::initializer_list< value_type > __list)
- [iterator](#) **insert** (value_type &&__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[iterator](#) **lower_bound** (const _Kt &__x)

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator lower_bound (const _Kt &__x) const`
- `iterator lower_bound (const key_type &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `multimap & operator= (const multimap &)=default`
- `multimap & operator= (initializer_list< value_type > __l)`
- `multimap & operator= (multimap &&)=default`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `reverse_iterator rend () noexcept`
- `void swap (multimap &__x) noexcept(/*conditional */)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator upper_bound (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator upper_bound (const _Kt &__x) const`
- `iterator upper_bound (const key_type &__x)`
- `const_iterator upper_bound (const key_type &__x) const`

Protected Member Functions

- `void _M_swap (_Safe_container &__x) noexcept`

Friends

- `template<typename _It, typename _SeqT, typename _CatT >
class ::__gnu_debug:: Safe_iterator`

6.721.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator =  
std::allocator<std::pair<const _Key, _Tp> >>  
class std::__debug::multimap<_Key, _Tp, _Compare, _Allocator >
```

Class std::multimap with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [multimap.h](#)

6.722 std::multimap<_Key, _Tp, _Compare, _Alloc > Class Template Reference

```
#include <stl_multimap.h>
```

Public Types

- `typedef _Alloc allocator_type`
- `typedef _Rep_type::const_iterator const_iterator`
- `typedef _Alloc_traits::const_pointer const_pointer`
- `typedef _Alloc_traits::const_reference const_reference`
- `typedef _Rep_type::const_reverse_iterator const_reverse_iterator`
- `typedef _Rep_type::difference_type difference_type`
- `typedef _Rep_type::iterator iterator`
- `typedef _Compare key_compare`
- `typedef _Key key_type`

- typedef `_Tp mapped_type`
- using `node_type` = typename `_Rep_type::node_type`
- typedef `_Alloc_traits::pointer pointer`
- typedef `_Alloc_traits::reference reference`
- typedef `_Rep_type::reverse_iterator reverse_iterator`
- typedef `_Rep_type::size_type size_type`
- typedef `std::pair< const_Key, _Tp > value_type`

Public Member Functions

- `multimap` ()=default
- template<typename `_InputIterator` >
`multimap` (`_InputIterator` __first, `_InputIterator` __last)
- template<typename `_InputIterator` >
`multimap` (`_InputIterator` __first, `_InputIterator` __last, const `_Compare` &__comp, const `allocator_type` &__a=allocator_type())
- template<typename `_InputIterator` >
`multimap` (`_InputIterator` __first, `_InputIterator` __last, const `allocator_type` &__a)
- `multimap` (const `_Compare` &__comp, const `allocator_type` &__a=allocator_type())
- `multimap` (const `allocator_type` &__a)
- `multimap` (const `multimap` &)=default
- `multimap` (const `multimap` &__m, const `_type_identity_t`< `allocator_type` > &__a)
- `multimap` (`initializer_list`< `value_type` > __l, const `_Compare` &__comp=_Compare(), const `allocator_type` &__a=allocator_type())
- `multimap` (`initializer_list`< `value_type` > __l, const `allocator_type` &__a)
- `multimap` (`multimap` &__m)=default
- `multimap` (`multimap` &&__m, const `_type_identity_t`< `allocator_type` > &__a) noexcept(is_nothrow_copy_constructible< `_Compare` >::value && `_Alloc_traits::S_always_equal`())
- `~multimap` ()=default
- const_iterator `begin` () const noexcept
- iterator `begin` () noexcept
- const_iterator `cbegin` () const noexcept
- const_iterator `cend` () const noexcept
- void `clear` () noexcept
- const_reverse_iterator `crbegin` () const noexcept
- const_reverse_iterator `crend` () const noexcept
- template<typename... `_Args` >
iterator `emplace` (`_Args` &&... __args)
- template<typename... `_Args` >
iterator `emplace_hint` (const_iterator __pos, `_Args` &&... __args)
- bool `empty` () const noexcept
- const_iterator `end` () const noexcept
- iterator `end` () noexcept
- size_type `erase` (const key_type &__x)
- iterator `erase` (const_iterator __first, const_iterator __last)
- node_type `extract` (const key_type &__x)
- node_type `extract` (const_iterator __pos)
- allocator_type `get_allocator` () const noexcept
- template<typename `_InputIterator` >
void `insert` (`_InputIterator` __first, `_InputIterator` __last)
- iterator `insert` (const_iterator __hint, node_type &&__nh)
- void `insert` (`initializer_list`< `value_type` > __l)

- iterator [insert](#) (node_type &&__nh)
 - key_compare [key_comp](#) () const
 - size_type [max_size](#) () const noexcept
 - template<typename _Cmp2 >
void [merge](#) ([map](#)<_Key, _Tp, _Cmp2, _Alloc> &&__source)
 - template<typename _Cmp2 >
void [merge](#) ([map](#)<_Key, _Tp, _Cmp2, _Alloc> &__source)
 - template<typename _Cmp2 >
void [merge](#) ([multimap](#)<_Key, _Tp, _Cmp2, _Alloc> &&__source)
 - template<typename _Cmp2 >
void [merge](#) ([multimap](#)<_Key, _Tp, _Cmp2, _Alloc> &__source)
 - [multimap](#) & [operator=](#) (const [multimap](#) &)=default
 - [multimap](#) & [operator=](#) (initializer_list<value_type> &__l)
 - [multimap](#) & [operator=](#) ([multimap](#) &&)=default
 - [const_reverse_iterator](#) [rbegin](#) () const noexcept
 - [reverse_iterator](#) [rbegin](#) () noexcept
 - [const_reverse_iterator](#) [rend](#) () const noexcept
 - [reverse_iterator](#) [rend](#) () noexcept
 - size_type [size](#) () const noexcept
 - void [swap](#) ([multimap](#) &__x) noexcept(*/*conditional */*)
 - value_compare [value_comp](#) () const
-
- iterator [insert](#) (const value_type &__x)
 - iterator [insert](#) (value_type &&__x)
 - template<typename _Pair >
__enable_if_t<[is_constructible](#)<value_type, _Pair>::value, iterator> [insert](#) (_Pair &&__x)
-
- iterator [insert](#) (const_iterator __position, const value_type &__x)
 - iterator [insert](#) (const_iterator __position, value_type &&__x)
 - template<typename _Pair >
__enable_if_t<[is_constructible](#)<value_type, _Pair &&>::value, iterator> [insert](#) (const_iterator __position, _Pair &&__x)
-
- iterator [erase](#) (const_iterator __position)
 - _GLIBCXX_ABI_TAG_CXX11 iterator [erase](#) (iterator __position)
-
- iterator [find](#) (const key_type &__x)
 - template<typename _Kt >
auto [find](#) (const _Kt &__x) -> decltype(_M_t._M_find_tr(__x))
-
- const_iterator [find](#) (const key_type &__x) const
 - template<typename _Kt >
auto [find](#) (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x))

- `size_type count` (const key_type &__x) const
- `template<typename _Kt >`
`auto count` (const _Kt &__x) const -> `decltype(_M_t._M_count_tr(__x))`
- `bool contains` (const key_type &__x) const
- `template<typename _Kt >`
`auto contains` (const _Kt &__x) const -> `decltype(_M_t._M_find_tr(__x), void(), true)`
- `iterator lower_bound` (const key_type &__x)
- `template<typename _Kt >`
`auto lower_bound` (const _Kt &__x) -> `decltype(iterator(_M_t._M_lower_bound_tr(__x)))`
- `const_iterator lower_bound` (const key_type &__x) const
- `template<typename _Kt >`
`auto lower_bound` (const _Kt &__x) const -> `decltype(const_iterator(_M_t._M_lower_bound_tr(__x)))`
- `iterator upper_bound` (const key_type &__x)
- `template<typename _Kt >`
`auto upper_bound` (const _Kt &__x) -> `decltype(iterator(_M_t._M_upper_bound_tr(__x)))`
- `const_iterator upper_bound` (const key_type &__x) const
- `template<typename _Kt >`
`auto upper_bound` (const _Kt &__x) const -> `decltype(const_iterator(_M_t._M_upper_bound_tr(__x)))`
- `std::pair< iterator, iterator > equal_range` (const key_type &__x)
- `template<typename _Kt >`
`auto equal_range` (const _Kt &__x) -> `decltype(pair< iterator, iterator >(_M_t._M_equal_range_tr(__x)))`
- `std::pair< const_iterator, const_iterator > equal_range` (const key_type &__x) const
- `template<typename _Kt >`
`auto equal_range` (const _Kt &__x) const -> `decltype(pair< const_iterator, const_iterator >(_M_t._M_equal_range_tr(__x)))`

Friends

- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator<` (const `multimap`< _K1, _T1, _C1, _A1 > &, const `multimap`< _K1, _T1, _C1, _A1 > &)
- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator==` (const `multimap`< _K1, _T1, _C1, _A1 > &, const `multimap`< _K1, _T1, _C1, _A1 > &)
- `template<typename, typename >`
`struct std::_Rb_tree_merge_helper`

6.722.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::←  
::allocator<std::pair<const _Key, _Tp> >>  
class std::multimap<_Key,_Tp,_Compare,_Alloc>
```

A standard container made up of (key,value) pairs, which can be retrieved based on a key, in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using equivalent keys). For a `multimap<Key, T>` the `key_type` is `Key`, the `mapped_type` is `T`, and the `value_type` is `std::pair<const Key, T>`.

Multimaps support bidirectional iterators.

The private tree data is declared exactly the same way for `map` and `multimap`; the distinction is made entirely in how the tree functions are called (`*_unique` versus `*_equal`, same as the standard).

6.722.2 Constructor & Destructor Documentation

multimap() [1/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap ( ) [default]
```

Default constructor creates no elements.

multimap() [2/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Creates a `multimap` with no elements.

Parameters

<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

multimap() [3/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    const multimap<_Key, _Tp, _Compare, _Alloc> & ) [default]
```

Multimap copy constructor.

Whether the allocator is copied depends on the allocator traits.

multimap() [4/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    multimap<_Key, _Tp, _Compare, _Alloc> && ) [default]
```

Multimap move constructor.

The newly-created multimap contains the exact contents of the moved instance. The moved instance is a valid, but unspecified multimap.

multimap() [5/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    initializer_list< value_type > __l,
    const _Compare & __comp = _Compare(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a multimap from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multimap consisting of copies of the elements from the initializer_list. This is linear in N if the list is already sorted, and NlogN otherwise (where N is `__l.size()`).

multimap() [6/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    const allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

multimap() [7/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    const multimap<_Key, _Tp, _Compare, _Alloc> & __m,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Allocator-extended copy constructor.

multimap() [8/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
    multimap<_Key, _Tp, _Compare, _Alloc> && __m,
    const __type_identity_t< allocator_type > & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

multimap() [9/12]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>
std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
```



```

        initializer_list< value_type > __l,
        const allocator_type & __a ) [inline]

```

Allocator-extended initializer-list constructor.

multimap() [10/12]

```

template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::multimap< _Key, _Tp, _Compare, _Alloc >::multimap (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a ) [inline]

```

Allocator-extended range constructor.

multimap() [11/12]

```

template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::multimap< _Key, _Tp, _Compare, _Alloc >::multimap (
    _InputIterator __first,
    _InputIterator __last ) [inline]

```

Builds a multimap from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a multimap consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

multimap() [12/12]

```

template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
std::multimap< _Key, _Tp, _Compare, _Alloc >::multimap (
    _InputIterator __first,
    _InputIterator __last,
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline]

```

Builds a multimap from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multimap consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already

sorted, and NlogN otherwise (where N is distance(__first,__last)).

~multimap()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
std::multimap<_Key, _Tp, _Compare, _Alloc >::~~multimap ( ) [default]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

6.722.3 Member Function Documentation

begin() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

begin() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

cbegin()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

cend()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

clear()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
void std::multimap<_Key, _Tp, _Compare, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in a multimap. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x), void()), true)    [inline]
```

Finds whether an element with the given key exists.

Parameters

\leftrightarrow	Key of (key, value) pairs to be located.
$_X$	

Returns

True if there is any element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
bool std::multimap< _Key, _Tp, _Compare, _Alloc >::contains (
    const key_type & __x ) const    [inline]
```

Finds whether an element with the given key exists.

Parameters

\leftrightarrow	Key of (key, value) pairs to be located.
$_X$	

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::count (
    const _Kt & __x ) const -> decltype(_M_t._M_count_tr(__x))    [inline]
```

Finds the number of elements with given key.

Parameters

\leftrightarrow	Key of (key, value) pairs to be located.
$_X$	

Returns

Number of elements with specified key.

count() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
```

```
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::multimap<_Key, _Tp, _Compare, _Alloc>::count (
    const key_type & __x ) const [inline]
```

Finds the number of elements with given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Number of elements with specified key.

crbegin()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::multimap<_Key, _Tp, _Compare, _Alloc>::crbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

crend()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::multimap<_Key, _Tp, _Compare, _Alloc>::crend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

emplace()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename... _Args>
iterator std::multimap<_Key, _Tp, _Compare, _Alloc>::emplace (
    _Args &&... __args ) [inline]
```

Build and insert a std::pair into the multimap.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).
---------------------	---

Returns

An iterator that points to the inserted (key,value) pair.

This function builds and inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

emplace_hint()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename... _Args>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Builds and inserts a `std::pair` into the multimap.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a `std::map` the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

empty()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
bool std::multimap< _Key, _Tp, _Compare, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the multimap is empty.

end() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

end() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::end ( ) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

Referenced by `std::multimap< _Key, _Tp, _Compare, _Alloc >::extract()`.

equal_range() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(pair<iterator, iterator>(_M_t._M_equal_range_tr(__x)))
[inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

equal_range() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(pair<const_iterator, const_iterator>(_M_t._M_
equal_range_tr(__x))) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

equal_range() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::pair< iterator, iterator > std::multimap< _Key, _Tp, _Compare, _Alloc >::equal_range (
    const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

equal_range() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
std::pair< const_iterator, const_iterator > std::multimap< _Key, _Tp, _Compare, _Alloc >::equal←
_range (
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

← _X	Key of (key, value) pairs to be located.
-----------------------	--

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

erase() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

← _X	Key of element to be erased.
-----------------------	------------------------------

Returns

The number of elements erased.

This function erases all elements located by the given key from a multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [first,last) range of elements from a multimap.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased .

Returns

The iterator `__last`.

This function erases a sequence of elements from a multimap. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from a multimap.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
_GLIBCXX_ABI_TAG_CXX11 iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from a multimap.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
node_type std::multimap< _Key, _Tp, _Compare, _Alloc >::extract (
    const key_type & __x ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
node_type std::multimap< _Key, _Tp, _Compare, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::multimap< _Key, _Tp, _Compare, _Alloc >::end()`.

find() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::find (
    const _Kt & __x ) -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::find (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::find (
    const key_type & __x ) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

get_allocator()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
```

```
std::allocator<std::pair<const _Key, _Tp> >>
allocator_type std::multimap< _Key, _Tp, _Compare, _Alloc >::get_allocator ( ) const [inline],
[noexcept]
```

Get a copy of the memory allocation object.

insert() [1/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _InputIterator >
void std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair >::value, iterator > std::multimap< _Key, _↵
Tp, _Compare, _Alloc >::insert (
    _Pair && __x ) [inline]
```

Inserts a `std::pair` into the multimap.

Parameters

<code>__↵ __x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
--------------------------	---

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a `std::map` the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

insert() [3/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Inserts a `std::pair` into the multimap.

Parameters

<code>↔</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
<code>_X</code>	

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

Referenced by `std::multimap< _Key, _Tp, _Compare, _Alloc >::insert()`.

insert() [4/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References `std::move()`.

insert() [5/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > std::multimap< _Key,
_Tp, _Compare, _Alloc >::insert (
    const_iterator __position,
    _Pair && __x ) [inline]
```

Inserts a std::pair into the multimap.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

insert() [6/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
```

```
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __position,
    const value_type & __x ) [inline]
```

Inserts a std::pair into the multimap.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

insert() [7/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    const_iterator __position,
    value_type && __x ) [inline]
```

Inserts a std::pair into the multimap.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

References [std::move\(\)](#).

insert() [8/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
void std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of std::pairs into the multimap.

Parameters

↩	A std::initializer_list<value_type> of pairs to be inserted.
↩	
↩	
↩	
/	

Complexity similar to that of the range constructor.

References [std::multimap< _Key, _Tp, _Compare, _Alloc >::insert\(\)](#).

insert() [9/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [10/10]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
    value_type && __x ) [inline]
```

Inserts a std::pair into the multimap.

Parameters

↩	Pair to be inserted (see std::make_pair for easy creation of pairs).
_X	

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

References [std::move\(\)](#).

key_comp()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
key_compare std::multimap< _Key, _Tp, _Compare, _Alloc >::key_comp ( ) const [inline]
```

Returns the key comparison object out of which the multimap was constructed.

lower_bound() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
```

```
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_lower_bound_tr(__x)))    [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) const -> decltype(const_iterator(_M_t._M_lower_bound_tr(__x)))
[inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful the iterator will point to the next greatest element or, if no such greater element exists, to end().

lower_bound() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) const [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful the iterator will point to the next greatest element or, if no such greater element exists, to end().

max_size()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::multimap< _Key, _Tp, _Compare, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the multimap.

operator=() [1/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
multimap & std::multimap< _Key, _Tp, _Compare, _Alloc >::operator= (
    const multimap< _Key, _Tp, _Compare, _Alloc > & ) [default]
```

Multimap assignment operator.

Whether the allocator is copied depends on the allocator traits.

operator=() [2/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
multimap & std::multimap< _Key, _Tp, _Compare, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Multimap list assignment operator.

Parameters

↔	An initializer_list.
_↔	
↔	
_↔	
/	

This function fills a multimap with copies of the elements in the initializer list __l.

Note that the assignment completely changes the multimap and that the resulting multimap's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
multimap & std::multimap< _Key, _Tp, _Compare, _Alloc >::operator= (
    multimap< _Key, _Tp, _Compare, _Alloc > && ) [default]
```

Move assignment operator.

rbegin() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::rbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

rbegin() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
reverse_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::rbegin ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

rend() [1/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_reverse_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::rend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

rend() [2/2]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
reverse_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::rend ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

size()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
size_type std::multimap<_Key, _Tp, _Compare, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the size of the multimap.

swap()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
void std::multimap<_Key, _Tp, _Compare, _Alloc >::swap (
    multimap<_Key, _Tp, _Compare, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another multimap.

Parameters

<code>__x</code>	A multimap of the same element and allocator types.
------------------	---

This exchanges the elements between two multimaps in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(m1,m2)` will feed to this function. Whether the allocators are swapped depends on the allocator traits.

upper_bound() [1/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap<_Key, _Tp, _Compare, _Alloc >::upper_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_upper_bound_tr(__x))) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns



Iterator pointing to the first element greater than key, or end().

upper_bound() [2/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
template<typename _Kt >
auto std::multimap<_Key, _Tp, _Compare, _Alloc >::upper_bound (
    const _Kt & __x ) const -> decltype(const_iterator(_M_t._M_upper_bound_tr(__x)))
[inline]
```

Finds the end of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
	

Returns



Read-only (constant) iterator pointing to first iterator greater than key, or end().

upper_bound() [3/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::upper_bound (
    const key_type & __x ) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
	

Returns



Iterator pointing to the first element greater than key, or end().

upper_bound() [4/4]

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::upper_bound (
    const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
	

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

value_comp()

```
template<typename _Key , typename _Tp , typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >>
value_compare std::multimap< _Key, _Tp, _Compare, _Alloc >::value_comp ( ) const [inline]
```

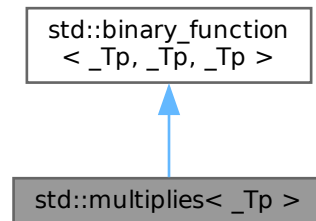
Returns a value comparison object, built from the key comparison object out of which the multimap was constructed.
The documentation for this class was generated from the following files:

- [stl_map.h](#)
- [stl_multimap.h](#)

6.723 std::multiplies<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::multiplies<_Tp>:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `_Tp` **operator()** (const `_Tp` &__x, const `_Tp` &__y) const

6.723.1 Detailed Description

```
template<typename _Tp>
struct std::multiplies<_Tp>
```

One of the [math functors](#).

6.723.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::second_argument_type [inherited]
second_argument_type is the type of the second argument
The documentation for this struct was generated from the following file:
```

- [stl_function.h](#)

6.724 `std::multiplies< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- template<typename `_Tp`, typename `_Up` >
constexpr auto **operator()** (`_Tp` &&`_t`, `_Up` &&`_u`) const noexcept(noexcept(`std::forward< _Tp >(_t)`
`*std::forward< _Up >(_u)`)) -> decltype(`std::forward< _Tp >(_t) *std::forward< _Up >(_u)`)

6.724.1 Detailed Description

One of the [math functors](#).

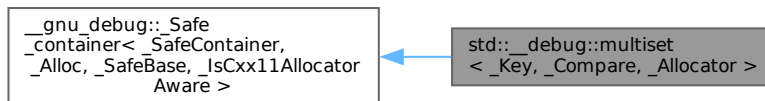
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.725 `std::__debug::multiset< _Key, _Compare, _Allocator >` Class Template Reference

```
#include <multiset.h>
```

Inheritance diagram for `std::__debug::multiset< _Key, _Compare, _Allocator >`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< _Base_const_iterator, multiset >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator< _Base_iterator, multiset >` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- using **node_type** = typename `_Base::node_type`
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Compare` **value_compare**
- typedef `_Key` **value_type**

Public Member Functions

- **multiset** ([_Base_ref](#) __x)
- `template<typename _InputIterator>`
multiset ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp=[_Compare](#)(), const [_Allocator](#) &__a=[_Allocator](#)())
- `template<typename _InputIterator>`
multiset ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- **multiset** (const [_Compare](#) &__comp, const [_Allocator](#) &__a=[_Allocator](#)())
- **multiset** (const [allocator_type](#) &__a)
- **multiset** (const [multiset](#) &)=default
- **multiset** (const [multiset](#) &__m, const [__type_identity_t](#)< [allocator_type](#) > &__a)
- **multiset** ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__comp=[_Compare](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- **multiset** ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- **multiset** ([multiset](#) &&__m)=default
- **multiset** ([multiset](#) &&__m, const [__type_identity_t](#)< [allocator_type](#) > &__a) noexcept(noexcept([_Base](#)(std::move(←__m), __a)))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) **begin** () const noexcept
- [iterator](#) **begin** () noexcept
- [const_iterator](#) **cbegin** () const noexcept
- [const_iterator](#) **cend** () const noexcept
- void **clear** () noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- `template<typename... _Args>`
[iterator](#) **emplace** ([_Args](#) &&... __args)
- `template<typename... _Args>`
[iterator](#) **emplace_hint** ([const_iterator](#) __pos, [_Args](#) &&... __args)
- [const_iterator](#) **end** () const noexcept
- [iterator](#) **end** () noexcept
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const [_Kt](#) &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const [_Kt](#) &__x) const
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const [key_type](#) &__x)
- [std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const [key_type](#) &__x) const
- [_Base_iterator](#) **erase** ([_Base_const_iterator](#) __position)
- [size_type](#) **erase** (const [key_type](#) &__x)
- [_GLIBCXX_ABI_TAG_CXX11](#) [iterator](#) **erase** ([const_iterator](#) __first, [const_iterator](#) __last)
- [_GLIBCXX_ABI_TAG_CXX11](#) [iterator](#) **erase** ([const_iterator](#) __position)
- [node_type](#) **extract** (const [key_type](#) &__key)
- [node_type](#) **extract** ([const_iterator](#) __position)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[iterator](#) **find** (const [_Kt](#) &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[const_iterator](#) **find** (const [_Kt](#) &__x) const
- [iterator](#) **find** (const [key_type](#) &__x)
- [const_iterator](#) **find** (const [key_type](#) &__x) const

- `template<typename __InputIterator >`
`void insert (__InputIterator __first, __InputIterator __last)`
- `iterator insert (const value_type &__x)`
- `iterator insert (const_iterator __hint, node_type &&__nh)`
- `iterator insert (const_iterator __position, const value_type &__x)`
- `iterator insert (const_iterator __position, value_type &&__x)`
- `void insert (initializer_list< value_type > __l)`
- `iterator insert (node_type &&__nh)`
- `iterator insert (value_type &&__x)`
- `template<typename __Kt, typename __Req = typename __has_is_transparent<_Compare, __Kt>::type>`
`iterator lower_bound (const __Kt &__x)`
- `template<typename __Kt, typename __Req = typename __has_is_transparent<_Compare, __Kt>::type>`
`const_iterator lower_bound (const __Kt &__x) const`
- `iterator lower_bound (const key_type &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `multiset & operator= (const multiset &)=default`
- `multiset & operator= (initializer_list< value_type > __l)`
- `multiset & operator= (multiset &&)=default`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `reverse_iterator rend () noexcept`
- `void swap (multiset &__x) noexcept(/*conditional */)`
- `template<typename __Kt, typename __Req = typename __has_is_transparent<_Compare, __Kt>::type>`
`iterator upper_bound (const __Kt &__x)`
- `template<typename __Kt, typename __Req = typename __has_is_transparent<_Compare, __Kt>::type>`
`const_iterator upper_bound (const __Kt &__x) const`
- `iterator upper_bound (const key_type &__x)`
- `const_iterator upper_bound (const key_type &__x) const`

Protected Member Functions

- `void _M_swap (_Safe_container &__x) noexcept`

Friends

- `template<typename __ItT, typename __SeqT, typename __CatT >`
`class ::__gnu_debug::_Safe_iterator`

6.725.1 Detailed Description

`template<typename __Key, typename __Compare = std::less<__Key>, typename __Allocator = std::allocator<__Key>>`

`class std::__debug::multiset< __Key, __Compare, __Allocator >`

Class `std::multiset` with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [multiset.h](#)

6.726 `std::multiset< __Key, __Compare, __Alloc >` Class Template Reference

```
#include <stl_multiset.h>
```

Public Types

- typedef _Alloc **allocator_type**
- typedef _Rep_type::const_iterator **const_iterator**
- typedef _Alloc_traits::const_pointer **const_pointer**
- typedef _Alloc_traits::const_reference **const_reference**
- typedef _Rep_type::const_reverse_iterator **const_reverse_iterator**
- typedef _Rep_type::difference_type **difference_type**
- typedef _Rep_type::const_iterator **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- using **node_type** = typename _Rep_type::node_type
- typedef _Alloc_traits::pointer **pointer**
- typedef _Alloc_traits::reference **reference**
- typedef _Rep_type::const_reverse_iterator **reverse_iterator**
- typedef _Rep_type::size_type **size_type**
- typedef _Compare **value_compare**
- typedef _Key **value_type**

Public Member Functions

- **multiset** ()=default
- template<typename _InputIterator >
multiset (_InputIterator __first, _InputIterator __last)
- template<typename _InputIterator >
multiset (_InputIterator __first, _InputIterator __last, const _Compare &__comp, const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
multiset (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- **multiset** (const _Compare &__comp, const allocator_type &__a=allocator_type())
- **multiset** (const allocator_type &__a)
- **multiset** (const **multiset** &)=default
- **multiset** (const **multiset** &__m, const __type_identity_t< allocator_type > &__a)
- **multiset** (**initializer_list**< value_type > __l, const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- **multiset** (**initializer_list**< value_type > __l, const allocator_type &__a)
- **multiset** (**multiset** &&)=default
- **multiset** (**multiset** &&__m, const __type_identity_t< allocator_type > &__a) noexcept(**is_nothrow_copy_constructible**< _Compare >::value && _Alloc_traits::_S_always_equal())
- **~multiset** ()=default
- iterator **begin** () const noexcept
- iterator **cbegin** () const noexcept
- iterator **cend** () const noexcept
- void **clear** () noexcept
- **reverse_iterator** **crbegin** () const noexcept
- **reverse_iterator** **crend** () const noexcept
- template<typename... _Args>
iterator **emplace** (_Args &&... __args)
- template<typename... _Args>
iterator **emplace_hint** (const_iterator __pos, _Args &&... __args)
- bool **empty** () const noexcept
- iterator **end** () const noexcept

- size_type **erase** (const key_type &__x)
 - _GLIBCXX_ABI_TAG_CXX11 iterator **erase** (const_iterator __first, const_iterator __last)
 - _GLIBCXX_ABI_TAG_CXX11 iterator **erase** (const_iterator __position)
 - node_type **extract** (const key_type &__x)
 - node_type **extract** (const_iterator __pos)
 - allocator_type **get_allocator** () const noexcept
 - template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
 - iterator **insert** (const value_type &__x)
 - iterator **insert** (const_iterator __hint, node_type &&__nh)
 - iterator **insert** (const_iterator __position, const value_type &__x)
 - iterator **insert** (const_iterator __position, value_type &&__x)
 - void **insert** (initializer_list< value_type > __l)
 - iterator **insert** (node_type &&__nh)
 - iterator **insert** (value_type &&__x)
 - key_compare **key_comp** () const
 - size_type **max_size** () const noexcept
 - template<typename _Compare1 >
void **merge** (multiset< _Key, _Compare1, _Alloc > &&__source)
 - template<typename _Compare1 >
void **merge** (multiset< _Key, _Compare1, _Alloc > &__source)
 - template<typename _Compare1 >
void **merge** (set< _Key, _Compare1, _Alloc > &&__source)
 - template<typename _Compare1 >
void **merge** (set< _Key, _Compare1, _Alloc > &__source)
 - multiset & **operator=** (const multiset &)=default
 - multiset & **operator=** (initializer_list< value_type > __l)
 - multiset & **operator=** (multiset &&)=default
 - reverse_iterator **rbegin** () const noexcept
 - reverse_iterator **rend** () const noexcept
 - size_type **size** () const noexcept
 - void **swap** (multiset &__x) noexcept(*/*conditional */*)
 - value_compare **value_comp** () const
-
- size_type **count** (const key_type &__x) const
 - template<typename _Kt >
auto **count** (const _Kt &__x) const -> decltype(_M_t._M_count_tr(__x))
-
- bool **contains** (const key_type &__x) const
 - template<typename _Kt >
auto **contains** (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x), void(), true)
-
- iterator **find** (const key_type &__x)
 - const_iterator **find** (const key_type &__x) const
 - template<typename _Kt >
auto **find** (const _Kt &__x) -> decltype(iterator{ _M_t._M_find_tr(__x) })
 - template<typename _Kt >
auto **find** (const _Kt &__x) const -> decltype(const_iterator{ _M_t._M_find_tr(__x) })

- iterator [lower_bound](#) (const key_type &__x)
- const_iterator [lower_bound](#) (const key_type &__x) const
- template<typename _Kt >
auto [lower_bound](#) (const _Kt &__x) -> decltype(iterator(_M_t._M_lower_bound_tr(__x)))
- template<typename _Kt >
auto [lower_bound](#) (const _Kt &__x) const -> decltype(iterator(_M_t._M_lower_bound_tr(__x)))
- iterator [upper_bound](#) (const key_type &__x)
- const_iterator [upper_bound](#) (const key_type &__x) const
- template<typename _Kt >
auto [upper_bound](#) (const _Kt &__x) -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))
- template<typename _Kt >
auto [upper_bound](#) (const _Kt &__x) const -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))
- std::pair< iterator, iterator > [equal_range](#) (const key_type &__x)
- std::pair< const_iterator, const_iterator > [equal_range](#) (const key_type &__x) const
- template<typename _Kt >
auto [equal_range](#) (const _Kt &__x) -> decltype(pair< iterator, iterator >(_M_t._M_equal_range_tr(__x)))
- template<typename _Kt >
auto [equal_range](#) (const _Kt &__x) const -> decltype(pair< iterator, iterator >(_M_t._M_equal_range_tr(__x)))

Friends

- template<typename _K1, typename _C1, typename _A1 >
bool **operator**< (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)
- template<typename _K1, typename _C1, typename _A1 >
bool **operator**== (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)
- template<typename, typename >
struct **std::**[Rb_tree_merge_helper](#)

6.726.1 Detailed Description

template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
class std::multiset< _Key, _Compare, _Alloc >

A standard container made up of elements, which can be retrieved in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using equivalent keys). For a `multiset<Key>` the `key_type` and `value_type` are `Key`.

Multisets support bidirectional iterators.

The private tree data is declared exactly the same way for set and multiset; the distinction is made entirely in how the tree functions are called (*_unique versus *_equal, same as the standard).

6.726.2 Constructor & Destructor Documentation

multiset() [1/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
std::multiset< _Key, _Compare, _Alloc >::multiset ( ) [default]
```

Default constructor creates no elements.

multiset() [2/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
std::multiset< _Key, _Compare, _Alloc >::multiset (
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Creates a multiset with no elements.

Parameters

<code>__comp</code>	Comparator to use.
<code>__a</code>	An allocator object.

multiset() [3/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
template<typename _InputIterator >
std::multiset< _Key, _Compare, _Alloc >::multiset (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Builds a multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a multiset consisting of copies of the elements from [first,last). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(__first,__last)).

multiset() [4/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
template<typename _InputIterator >
std::multiset< _Key, _Compare, _Alloc >::multiset (
    _InputIterator __first,
    _InputIterator __last,
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multiset consisting of copies of the elements from `[__first,__last)`. This is linear in N if the range is already sorted, and NlogN otherwise (where N is `distance(__first,__last)`).

multiset() [5/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    const multiset< _Key, _Compare, _Alloc > & ) [default]
```

Multiset copy constructor.

Whether the allocator is copied depends on the allocator traits.

multiset() [6/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    multiset< _Key, _Compare, _Alloc > && ) [default]
```

Multiset move constructor.

The newly-created multiset contains the exact contents of the moved instance. The moved instance is a valid, but unspecified multiset.

multiset() [7/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    initializer_list< value_type > __l,
    const _Compare & __comp = _Compare(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a multiset from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> .
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multiset consisting of copies of the elements from the list. This is linear in N if the list is already sorted, and NlogN otherwise (where N is `__l.size()`).

multiset() [8/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
```

```
const allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

multiset() [9/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    const multiset< _Key, _Compare, _Alloc > & __m,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Allocator-extended copy constructor.

multiset() [10/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    multiset< _Key, _Compare, _Alloc > && __m,
    const __type_identity_t< allocator_type > & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

multiset() [11/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
std::multiset< _Key, _Compare, _Alloc >::multiset (
    initializer_list< value_type > __l,
    const allocator_type & __a ) [inline]
```

Allocator-extended initializer-list constructor.

multiset() [12/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _InputIterator >
std::multiset< _Key, _Compare, _Alloc >::multiset (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a ) [inline]
```

Allocator-extended range constructor.

~multiset()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
std::multiset< _Key, _Compare, _Alloc >::~multiset ( ) [default]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

6.726.3 Member Function Documentation

begin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the multiset. Iteration is done in ascending order according to the keys.

cbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
iterator std::multiset< _Key, _Compare, _Alloc >::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the multiset. Iteration is done in ascending order according to the keys.

cend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
iterator std::multiset< _Key, _Compare, _Alloc >::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the multiset. Iteration is done in ascending order according to the keys.

clear()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
void std::multiset< _Key, _Compare, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in a multiset. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
template<typename _Kt >
```

```
auto std::multiset< _Key, _Compare, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x), void(), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_↵</code>	Key of elements to be located.
<code>_X</code>	

Returns

True if there is any element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
bool std::multiset< _Key, _Compare, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_↔</code>	Key of elements to be located.
<code>_X</code>	

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::count (
    const _Kt & __x ) const -> decltype(_M_t._M_count_tr(__x))    [inline]
```

Finds the number of elements with given key.

Parameters

<code>_↔</code>	Key of elements to be located.
<code>_X</code>	

Returns

Number of elements with specified key.

count() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
size_type std::multiset< _Key, _Compare, _Alloc >::count (
    const key_type & __x ) const    [inline]
```

Finds the number of elements with given key.

Parameters

<code>_↔</code>	Key of elements to be located.
<code>_X</code>	

Returns

Number of elements with specified key.

crbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::crbegin ( ) const    [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

crend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

emplace()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
template<typename... _Args>
```

```
iterator std::multiset< _Key, _Compare, _Alloc >::emplace (
    _Args &&... __args ) [inline]
```

Builds and inserts an element into the multiset.

Parameters

<code>__args</code>	Arguments used to generate the element instance to be inserted.
---------------------	---

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Insertion requires logarithmic time.

emplace_hint()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
template<typename... _Args>
```

```
iterator std::multiset< _Key, _Compare, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Builds and inserts an element into the multiset.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element instance to be inserted.

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

empty()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
bool std::multiset< _Key, _Compare, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the set is empty.

end()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
iterator std::multiset< _Key, _Compare, _Alloc >::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the multiset. Iteration is done in ascending order according to the keys.

Referenced by [std::multiset< _Key, _Compare, _Alloc >::extract\(\)](#).

equal_range() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
template<typename _Kt >
```

```
auto std::multiset< _Key, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(pair<iterator, iterator>(_M_t._M_equal_range_tr(__x)))
```

```
[inline]
```

Finds a subsequence matching given key.

Parameters

_Key	Key to be located.
_X	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
template<typename _Kt >
```

```
auto std::multiset< _Key, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(pair<iterator, iterator>(_M_t._M_equal_range_
tr(__x))) [inline]
```

Finds a subsequence matching given key.

Parameters

_Key	Key to be located.
_X	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
std::pair< iterator, iterator > std::multiset< _Key, _Compare, _Alloc >::equal_range (
    const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
std::pair< const_iterator, const_iterator > std::multiset< _Key, _Compare, _Alloc >::equal_range
(
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

erase() [1/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::multiset< _Key, _Compare, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all elements located by the given key from a multiset. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
_GLIBCXX_ABI_TAG_CXX11 iterator std::multiset< _Key, _Compare, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a `[first,last)` range of elements from a multiset.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator *last*.

This function erases a sequence of elements from a multiset. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
_GLIBCXX_ABI_TAG_CXX11 iterator std::multiset< _Key, _Compare, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from a multiset.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, end() is returned.

This function erases an element, pointed to by the given iterator, from a multiset. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
node_type std::multiset< _Key, _Compare, _Alloc >::extract (
    const key_type & __x ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
node_type std::multiset< _Key, _Compare, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References [std::multiset< _Key, _Compare, _Alloc >::end\(\)](#).

find() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::find (
    const _Kt & __x ) -> decltype(iterator{_M_t._M_find_tr(__x)}) [inline]
```

Tries to locate an element in a set.

Parameters

_↵	Element to be located.
_X	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::find (
    const _Kt & __x ) const -> decltype(const_iterator{_M_t._M_find_tr(__x)}) [inline]
```

Tries to locate an element in a set.

Parameters

<code>_↔</code>	Element to be located.
<code>_X</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::find (
    const key_type & __x ) [inline]
```

Tries to locate an element in a set.

Parameters

<code>_↔</code>	Element to be located.
<code>_X</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
const_iterator std::multiset< _Key, _Compare, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in a set.

Parameters

<code>_↔</code>	Element to be located.
<code>_X</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

get_allocator()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
```

allocator_type [std::multiset](#)< _Key, _Compare, _Alloc >::get_allocator () const [inline], [noexcept]
Returns the memory allocation object.

insert() [1/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _InputIterator >
void std::multiset< _Key, _Compare, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that tries to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Inserts an element into the multiset.

Parameters

<code>↵ __x</code>	Element to be inserted.
------------------------	-------------------------

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Insertion requires logarithmic time.

Referenced by [std::multiset< _Key, _Compare, _Alloc >::insert\(\)](#).

insert() [3/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [4/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
```

```

_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::insert (
    const_iterator __position,
    const value_type & __x ) [inline]

```

Inserts an element into the multiset.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

insert() [5/6]

```

template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
void std::multiset< _Key, _Compare, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]

```

Attempts to insert a list of elements into the multiset.

Parameters

<code>↵</code>	A <code>std::initializer_list<value_type></code> of elements to be inserted.
<code>↵</code>	
<code>↵</code>	
<code>↵</code>	
<code>/</code>	

Complexity similar to that of the range constructor.

References [std::multiset< _Key, _Compare, _Alloc >::insert\(\)](#).

insert() [6/6]

```

template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::insert (
    node_type && __nh ) [inline]

```

Re-insert an extracted node.

References [std::move\(\)](#).

key_comp()

```

template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>

```

```
key_compare std::multiset< _Key, _Compare, _Alloc >::key_comp ( ) const [inline]
```

Returns the comparison object.

lower_bound() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_lower_bound_tr(__x))) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_x</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) const -> decltype(iterator(_M_t._M_lower_bound_tr(__x))) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_x</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
const_iterator std::multiset< _Key, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) const [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

max_size()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
size_type std::multiset< _Key, _Compare, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the set.

operator=() [1/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
multiset & std::multiset< _Key, _Compare, _Alloc >::operator= (
    const multiset< _Key, _Compare, _Alloc > & ) [default]
```

Multiset assignment operator.

Whether the allocator is copied depends on the allocator traits.

operator=() [2/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
```

```
multiset & std::multiset< _Key, _Compare, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Multiset list assignment operator.

Parameters

↔	An initializer_list.
↔	
↔	
↔	
/	

This function fills a multiset with copies of the elements in the initializer list __l.

Note that the assignment completely changes the multiset and that the resulting multiset's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
multiset & std::multiset< _Key, _Compare, _Alloc >::operator= (
    multiset< _Key, _Compare, _Alloc > && ) [default]
```

Move assignment operator.

rbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

rend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

size()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
size_type std::multiset< _Key, _Compare, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the size of the set.

swap()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
void std::multiset< _Key, _Compare, _Alloc >::swap (
    multiset< _Key, _Compare, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another multiset.

Parameters

<code>_↔</code>	A multiset of the same element and allocator types.
<code>_X</code>	

This exchanges the elements between two multisets in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

upper_bound() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::upper_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))    [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
template<typename _Kt >
auto std::multiset< _Key, _Compare, _Alloc >::upper_bound (
    const _Kt & __x ) const -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))    [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↔
_Key>>
iterator std::multiset< _Key, _Compare, _Alloc >::upper_bound (
    const key_type & __x )    [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
const_iterator std::multiset< _Key, _Compare, _Alloc >::upper_bound (
    const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

value_comp()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
value_compare std::multiset< _Key, _Compare, _Alloc >::value_comp ( ) const [inline]
```

Returns the comparison object.

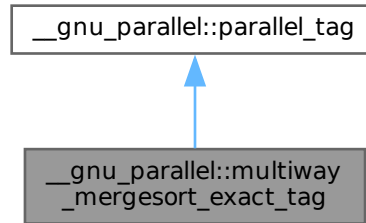
The documentation for this class was generated from the following file:

- [stl_multiset.h](#)

6.727 __gnu_parallel::multiway_mergesort_exact_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::multiway_mergesort_exact_tag`:



Public Member Functions

- `multiway_mergesort_exact_tag` ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) `__get_num_threads` ()
- void `set_num_threads` ([_ThreadIndex](#) __num_threads)

6.727.1 Detailed Description

Forces parallel sorting using multiway mergesort with exact splitting at compile time.

6.727.2 Member Function Documentation

`__get_num_threads()`

[_ThreadIndex](#) `__gnu_parallel::parallel_tag::__get_num_threads` () [inline], [inherited]

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

void `__gnu_parallel::parallel_tag::set_num_threads` ([_ThreadIndex](#) __num_threads) [inline], [inherited]

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

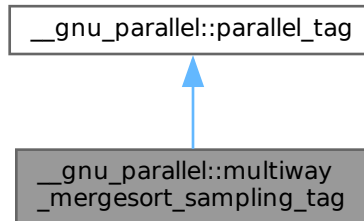
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.728 __gnu_parallel::multiway_mergesort_sampling_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::multiway_mergesort_sampling_tag:



Public Member Functions

- **multiway_mergesort_sampling_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **__get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

6.728.1 Detailed Description

Forces parallel sorting using multiway mergesort with splitting by sampling at compile time.

6.728.2 Member Function Documentation

__get_num_threads()

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#).

set_num_threads()

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

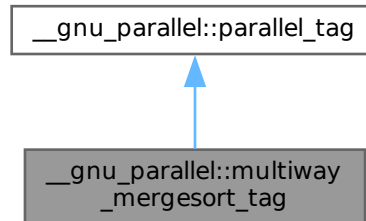
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.729 __gnu_parallel::multiway_mergesort_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::multiway_mergesort_tag:



Public Member Functions

- `multiway_mergesort_tag` (`_ThreadIndex` __num_threads)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` __num_threads)

6.729.1 Detailed Description

Forces parallel sorting using multiway mergesort at compile time.

6.729.2 Member Function Documentation

`__get_num_threads()`

```
_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

```
void __gnu_parallel::parallel_tag::set_num_threads (
    _ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.730 std::mutex Class Reference

```
#include <std_mutex.h>
Inherits std::__mutex_base.
```

Public Types

- typedef __native_type * **native_handle_type**

Public Member Functions

- **mutex** (const [mutex](#) &)=delete
- void **lock** ()
- native_handle_type **native_handle** () noexcept
- [mutex](#) & **operator=** (const [mutex](#) &)=delete
- bool **try_lock** () noexcept
- void **unlock** ()

6.730.1 Detailed Description

The standard mutex type.

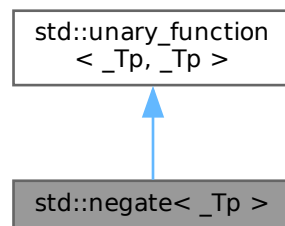
The documentation for this class was generated from the following file:

- [std_mutex.h](#)

6.731 std::negate<_Tp> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::negate<_Tp>:



Public Types

- typedef _Tp [argument_type](#)
- typedef _Tp [result_type](#)

Public Member Functions

- constexpr _Tp **operator()** (const _Tp &__x) const

6.731.1 Detailed Description

```
template<typename _Tp>
struct std::negate< _Tp >
```

One of the [math functors](#).

6.731.2 Member Typedef Documentation

argument_type

```
typedef _Tp std::unary_function< _Tp , _Tp >::argument_type [inherited]
argument_type is the type of the argument
```

result_type

```
typedef _Tp std::unary_function< _Tp , _Tp >::result_type [inherited]
result_type is the return type
```

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.732 `std::negate< void >` Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- template<typename _Tp >
constexpr auto **operator()** (_Tp &&__t) const noexcept(noexcept([std::forward](#)< _Tp >(__t))) -> decltype([std::forward](#)< _Tp >(__t))

6.732.1 Detailed Description

One of the [math functors](#).

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.733 `std::negative_binomial_distribution< _IntType >` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **negative_binomial_distribution** (`_IntType __k`, `double __p=0.5`)
- **negative_binomial_distribution** (`const param_type &__p`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `template<typename _UniformRandomNumberGenerator >`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `_IntType k () const`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `double p () const`
- `param_type param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::negative_binomial_distribution< _IntType1 > &__x)`
- `bool operator== (const negative_binomial_distribution &__d1, const negative_binomial_distribution &__d2)`
- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::negative_binomial_distribution< _IntType1 > &__x)`

6.733.1 Detailed Description

```
template<typename _IntType = int>
class std::negative_binomial_distribution< _IntType >
```

A `negative_binomial_distribution` random number distribution.

The formula for the negative binomial probability mass function is $p(i) = \binom{n}{i} p^i (1-p)^{t-i}$ where t and p are the parameters of the distribution.

6.733.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::negative_binomial_distribution< _IntType >::result_type
```

The type of the range of the distribution.

6.733.3 Member Function Documentation

k()

```
template<typename _IntType = int>
_IntType std::negative_binomial_distribution< _IntType >::k ( ) const [inline]
```

Return the k parameter of the distribution.

max()

```
template<typename _IntType = int>
result_type std::negative_binomial_distribution< _IntType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits< _Tp >::max\(\)](#).

min()

```
template<typename _IntType = int>
result_type std::negative_binomial_distribution< _IntType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>()()

```
template<typename _IntType >
template<typename _UniformRandomNumberGenerator >
negative_binomial_distribution< _IntType >::result_type std::negative_binomial_distribution< _↵
_IntType >::operator() (
    _UniformRandomNumberGenerator & __urng )
```

Generating functions.

p()

```
template<typename _IntType = int>
double std::negative_binomial_distribution< _IntType >::p ( ) const [inline]
```

Return the p parameter of the distribution.

param() [1/2]

```
template<typename _IntType = int>
param_type std::negative_binomial_distribution< _IntType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _IntType = int>
void std::negative_binomial_distribution< _IntType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _IntType = int>
void std::negative_binomial_distribution< _IntType >::reset ( ) [inline]
```

Resets the distribution state.

References [std::gamma_distribution< _RealType >::reset\(\)](#).

6.733.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::negative_binomial_distribution< _IntType1 > & __x ) [friend]
```

Inserts a `negative_binomial_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>negative_binomial_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _IntType = int>
bool operator== (
    const negative_binomial_distribution< _IntType > & __d1,
    const negative_binomial_distribution< _IntType > & __d2 ) [friend]
```

Return true if two negative binomial distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::negative_binomial_distribution< _IntType1 > & __x ) [friend]
```

Extracts a `negative_binomial_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>negative_binomial_distribution</code> random number generator engine.

Returns

The input stream with `___x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.734 `std::nested_exception` Class Reference

```
#include <nested_exception.h>
```

Public Member Functions

- `nested_exception` (const [nested_exception](#) &) noexcept=default
- `exception_ptr nested_ptr` () const noexcept
- `nested_exception & operator=` (const [nested_exception](#) &) noexcept=default
- `void rethrow_nested` () const

6.734.1 Detailed Description

Exception class with `exception_ptr` data member.

The documentation for this class was generated from the following file:

- [nested_exception.h](#)

6.735 `__gnu_cxx::limit_condition::never_adjustor` Struct Reference

```
#include <throw_allocator.h>
```

Inherits `__gnu_cxx::limit_condition::adjustor_base`.

6.735.1 Detailed Description

Never enter the condition.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.736 `__gnu_cxx::random_condition::never_adjustor` Struct Reference

```
#include <throw_allocator.h>
```

Inherits `__gnu_cxx::random_condition::adjustor_base`.

6.736.1 Detailed Description

Never enter the condition.

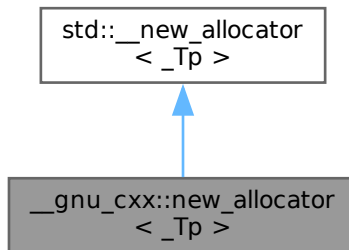
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.737 `__gnu_cxx::new_allocator<_Tp>` Class Template Reference

```
#include <new_allocator.h>
```

Inheritance diagram for `__gnu_cxx::new_allocator<_Tp>`:



Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef `std::true_type` **propagate_on_container_move_assignment**
- typedef `std::size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **new_allocator** (const `new_allocator` &) noexcept
- template<typename `_Tp1` >
 new_allocator (const `new_allocator`<`_Tp1`> &) noexcept
- `_Tp` * **allocate** (size_type __n, const void * =static_cast< const void * >(0))
- void **deallocate** (`_Tp` * __p, size_type __n)

6.737.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::new_allocator<_Tp>
```

An allocator that uses global new, as per C++03 [20.4.1].
This is precisely the allocator defined in the C++ Standard.

- all allocation calls operator new
- all deallocation calls operator delete

Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

The documentation for this class was generated from the following file:

- [ext/new_allocator.h](#)

6.738 `__gnu_pbds::detail::no_throw_copies< Key, Mapped >` Struct Template Reference

```
#include <types_traits.hpp>
```

Public Types

- `typedef integral_constant< int, __simple > indicator`

Static Public Attributes

- `static const bool __simple`

6.738.1 Detailed Description

```
template<typename Key, typename Mapped>
struct __gnu_pbds::detail::no_throw_copies< Key, Mapped >
```

Primary template.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.739 `__gnu_pbds::detail::no_throw_copies< Key, null_type >` Struct Template Reference

```
#include <types_traits.hpp>
```

Public Types

- `typedef integral_constant< int, is_simple< Key >::value > indicator`

6.739.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::no_throw_copies< Key, null_type >
```

Specialization.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.740 `std::normal_distribution< _RealType >` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- `typedef _RealType result_type`

Public Member Functions

- **normal_distribution** (const [param_type](#) &__p)
- **normal_distribution** ([result_type](#) __mean, [result_type](#) __stddev=[result_type](#)(1))
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- _RealType **mean** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()
- _RealType **stddev** () const

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::normal_distribution](#)< _RealType1 > &__x)
- template<typename _RealType1 >
bool **operator==** (const [std::normal_distribution](#)< _RealType1 > &__d1, const [std::normal_distribution](#)< _RealType1 > &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::normal_distribution](#)< _RealType1 > &__x)

6.740.1 Detailed Description

```
template<typename _RealType = double>
class std::normal_distribution< _RealType >
```

A normal continuous distribution for random numbers.
The formula for the normal probability density function is

$$p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu}{2\sigma^2}}$$

6.740.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::normal\_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.740.3 Constructor & Destructor Documentation

normal_distribution()

```
template<typename _RealType = double>
std::normal_distribution< _RealType >::normal_distribution (
    result_type __mean,
    result_type __stddev = result_type(1) ) [inline], [explicit]
```

Constructs a normal distribution with parameters *mean* and standard deviation.

6.740.4 Member Function Documentation

max()

```
template<typename _RealType = double>
result_type std::normal_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits<_Tp>::max\(\)](#).

mean()

```
template<typename _RealType = double>
_RealType std::normal_distribution< _RealType >::mean ( ) const [inline]
```

Returns the mean of the distribution.

min()

```
template<typename _RealType = double>
result_type std::normal_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.
References [std::numeric_limits<_Tp>::lowest\(\)](#).

operator>() [1/2]

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::normal_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::normal_distribution<_RealType>::operator\(\)](#).

Referenced by [std::normal_distribution<_RealType>::operator\(\)](#).

operator>() [2/2]

```
template<typename _RealType >
template<typename _UniformRandomNumberGenerator >
normal_distribution< _RealType >::result_type std::normal_distribution< _RealType >::operator()
(
    _UniformRandomNumberGenerator & __urng,
    const param_type & __param )
```

Polar method due to Marsaglia.

Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. V, Sect. 4.4.

References [std::log\(\)](#), and [std::sqrt\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::normal_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::normal_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::normal_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

Referenced by `std::lognormal_distribution< _RealType >::reset()`, `std::gamma_distribution< _RealType >::reset()`, `std::student_t_distribution< _RealType >::reset()`, `std::binomial_distribution< _IntType >::reset()`, and `std::poisson_distribution< _IntType >::reset()`.

stddev()

```
template<typename _RealType = double>
_RealType std::normal_distribution< _RealType >::stddev ( ) const [inline]
```

Returns the standard deviation of the distribution.

6.740.5 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::normal_distribution< _RealType1 > & __x ) [friend]
```

Inserts a normal_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A normal_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
```

```
template<typename _RealType1 >
bool operator== (
    const std::normal_distribution< _RealType1 > & __d1,
    const std::normal_distribution< _RealType1 > & __d2 ) [friend]
```



Return true if two normal distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::normal_distribution< _RealType1 > & __x ) [friend]
```

Extracts a normal_distribution random number distribution __x from the input stream __is.

Parameters

 <code>__is</code>	An input stream.
 <code>__x</code>	A normal_distribution random number generator engine.

Returns

The input stream with __x extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.741 std::nostopstate_t Struct Reference

6.741.1 Detailed Description

Tag type indicating a stop_source should have no shared-stop-state.

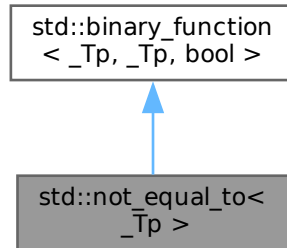
The documentation for this struct was generated from the following file:

- [stop_token](#)

6.742 std::not_equal_to< _Tp > Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::not_equal_to<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `bool` **operator()** (`const _Tp &__x`, `const _Tp &__y`) const

6.742.1 Detailed Description

template<typename _Tp>
struct `std::not_equal_to<_Tp>`

One of the [comparison functors](#).

6.742.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` [std::binary_function<_Tp, _Tp, bool>::first_argument_type](#) [inherited]
`first_argument_type` is the type of the first argument

result_type

typedef `bool` [std::binary_function<_Tp, _Tp, bool>::result_type](#) [inherited]
`result_type` is the return type

second_argument_type

typedef `_Tp` [std::binary_function<_Tp, _Tp, bool>::second_argument_type](#) [inherited]
`second_argument_type` is the type of the second argument

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.743 std::ranges::not_equal_to Struct Reference

```
#include <ranges_cmp.h>
```

Public Types

- using **is_transparent** = __is_transparent

Public Member Functions

- template<typename _Tp, typename _Up >
requires equality_comparable_with<_Tp, _Up>
constexpr bool **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::declval<_Up >()==std::declval<_Tp >()))

6.743.1 Detailed Description

ranges::not_equal_to function object type.

The documentation for this struct was generated from the following file:

- [ranges_cmp.h](#)

6.744 std::not_equal_to< void > Struct Reference

```
#include <stl_function.h>
```

Public Types

- typedef __is_transparent **is_transparent**

Public Member Functions

- template<typename _Tp, typename _Up >
constexpr auto **operator()** (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp >(__t) !=std::forward<_Up >(__u))) -> decltype(std::forward<_Tp >(__t) !=std::forward<_Up >(__u))

6.744.1 Detailed Description

One of the [comparison functors](#).

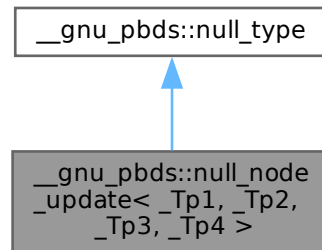
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.745 __gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 > Struct Template Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::null_node_update<_Tp1, _Tp2, _Tp3, _Tp4>`:



6.745.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Tp3, typename _Tp4>
struct __gnu_pbds::null_node_update<_Tp1, _Tp2, _Tp3, _Tp4>
```

A null node updatator, indicating that no node updates are required.

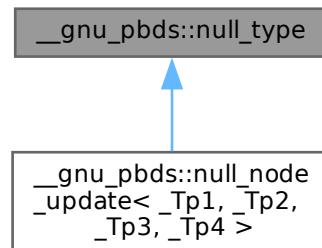
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.746 __gnu_pbds::null_type Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::null_type`:



6.746.1 Detailed Description

Represents no type, or absence of type, for template tricks.

In a mapped-policy, indicates that an associative container is a set.

In a list-update policy, indicates that each link does not need metadata.

In a hash policy, indicates that the combining hash function is actually a ranged hash function.

In a probe policy, indicates that the combining probe function is actually a ranged probe function.
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.747 `std::experimental::fundamentals_v1::nullopt_t` Struct Reference

Public Types

- enum class `_Construct` { `_Token` }

Public Member Functions

- constexpr `nullopt_t` (`_Construct`)

6.747.1 Detailed Description

Tag type to disengage optional objects.

The documentation for this struct was generated from the following file:

- [experimental/optional](#)

6.748 `std::nullopt_t` Struct Reference

Public Types

- enum class `_Construct` { `_Token` }

Public Member Functions

- constexpr `nullopt_t` (`_Construct`) noexcept

6.748.1 Detailed Description

Tag type to disengage optional objects.

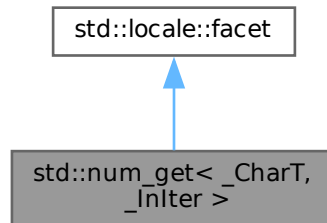
The documentation for this struct was generated from the following file:

- [optional](#)

6.749 `std::num_get<_CharT, _InIter>` Class Template Reference

```
#include <locale_facets.h>
```


Inheritance diagram for `std::num_get<_CharT, _InIter >`:



Public Types

- typedef `_CharT` `char_type`
- typedef `_InIter` `iter_type`

Public Member Functions

- `num_get` (`size_t __refs=0`)
- `template<typename _ValueT >`
`_InIter _M_extract_int` (`_InIter __beg, _InIter __end, ios_base &__io, ios_base::iostate &__err, _ValueT &__v`)
`const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, bool &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, void *&__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, long &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, unsigned short &__v`)
`const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, unsigned int &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, unsigned long &__v`)
`const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, long long &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, unsigned long long &__v`)
`const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, float &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, double &__v`) `const`
- `iter_type get` (`iter_type __in, iter_type __end, ios_base &__io, ios_base::iostate &__err, long double &__v`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual [~num_get](#) ()
- [iter_type _M_extract_float](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [string](#) &) const
- [template<typename _ValueT>](#)
[iter_type _M_extract_int](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [_ValueT](#) &) const
- [template<typename _CharT2>](#)
[__gnu_cxx::__enable_if<__is_char<_CharT2>::__value, int>::__type _M_find](#) (const [_CharT2](#) *, [size_t](#) __len, [_CharT2](#) __c) const
- [template<typename _CharT2>](#)
[__gnu_cxx::__enable_if<!__is_char<_CharT2>::__value, int>::__type _M_find](#) (const [_CharT2](#) *__zero, [size_t](#) __len, [_CharT2](#) __c) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [bool](#) &) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [long](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [unsigned short](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [unsigned int](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [unsigned long](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [long long](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, [unsigned long long](#) &__v) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [float](#) &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [double](#) &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [long double](#) &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base](#) &, [ios_base::iostate](#) &, [void](#) *&) const

Static Protected Member Functions

- static [__c_locale _S_clone_c_locale](#) ([__c_locale](#) &__cloc) throw ()
- static void [_S_create_c_locale](#) ([__c_locale](#) &__cloc, const char *__s, [__c_locale](#) __old=0)
- static void [_S_destroy_c_locale](#) ([__c_locale](#) &__cloc)
- static [__c_locale _S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static [__c_locale _S_lc_ctype_c_locale](#) ([__c_locale](#) __cloc, const char *__s)

6.749.1 Detailed Description

template<typename _CharT, typename _InIter>
class std::num_get<_CharT, _InIter>

Primary class template num_get.

This facet encapsulates the code to parse and return a number from a string. It is used by the istream numeric extraction operators.

The num_get template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the num_get facet.

6.749.2 Member Typedef Documentation

char_type

```
template<typename _CharT , typename _InIter >
typedef _CharT std::num_get< _CharT, _InIter >::char_type
```

Public typedefs.

iter_type

```
template<typename _CharT , typename _InIter >
typedef _InIter std::num_get< _CharT, _InIter >::iter_type
```

Public typedefs.

6.749.3 Constructor & Destructor Documentation

num_get()

```
template<typename _CharT , typename _InIter >
std::num_get< _CharT, _InIter >::num_get (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~num_get()

```
template<typename _CharT , typename _InIter >
virtual std::num_get< _CharT, _InIter >::~num_get ( ) [inline], [protected], [virtual]
```

Destructor.

6.749.4 Member Function Documentation

do_get() [1/11]

```
template<typename _CharT , typename _InIter >
virtual iter_type std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
--------------------	------------------------

Parameters

<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [2/11]

```
template<typename _CharT , typename _InIter >
virtual iter_type std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long long & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [3/11]

```
template<typename _CharT , typename _InIter >
virtual iter_type std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned int & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [4/11]

```
template<typename _CharT , typename _InIter >
virtual iter\_type std::num\_get< _CharT, _InIter >::do_get (
    iter\_type __beg,
    iter\_type __end,
    ios\_base & __io,
    ios\_base::iostate & __err,
    unsigned long & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [5/11]

```
template<typename _CharT , typename _InIter >
virtual iter\_type std::num\_get< _CharT, _InIter >::do_get (
    iter\_type __beg,
    iter\_type __end,
    ios\_base & __io,
```

```
ios_base::iostate & __err,
unsigned long long & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [6/11]

```
template<typename _CharT , typename _InIter >
virtual iter_type std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned short & __v ) const [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

do_get() [7/11]

```
template<typename _CharT , typename _InIter >
```

```

_InIter std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    bool & __v ) const [protected], [virtual]

```

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::ios_base::_M_getloc\(\)](#), [std::ios_base::boolalpha](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::flags\(\)](#), and [std::ios_base::goodbit](#).

Referenced by [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), [std::num_get< _CharT, _InIter >::get\(\)](#), and [std::num_get< _CharT, _InIter >::get\(\)](#).

do_get() [8/11]

```

template<typename _CharT , typename _InIter >
_InIter std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    double & __v ) const [protected], [virtual]

```

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::basic_string<_CharT, _Traits, _Alloc >::c_str\(\)](#), [std::ios_base::eofbit](#), and [std::basic_string<_CharT, _Traits, _Alloc >::r](#)

do_get() [9/11]

```
template<typename _CharT , typename _InIter >
_InIter std::num_get<_CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    float & __v ) const [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable v. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::basic_string<_CharT, _Traits, _Alloc >::c_str\(\)](#), [std::ios_base::eofbit](#), and [std::basic_string<_CharT, _Traits, _Alloc >::r](#)

do_get() [10/11]

```
template<typename _CharT , typename _InIter >
_InIter std::num_get<_CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long double & __v ) const [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable v. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__beg</code>	Start of input stream.
--------------------	------------------------

Parameters

<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::basic_string<_CharT, _Traits, _Alloc>::c_str\(\)](#), [std::ios_base::eofbit](#), and [std::basic_string<_CharT, _Traits, _Alloc>::r](#)

do_get() [11/11]

```
template<typename _CharT , typename _InIter >
_InIter std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    void *& __v ) const [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::ios_base::basefield](#), [std::ios_base::flags\(\)](#), and [std::ios_base::hex](#).

get() [1/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    bool & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the bool *v*. It does so by calling num_get::do_get().

If ios_base::boolalpha is set, attempts to read ctype<CharT>::trunename() or ctype<CharT>::falsename(). Sets *v* to true or false if successful. Sets err to ios_base::failbit if reading the string fails. Sets err to ios_base::eofbit if the stream is emptied.

If ios_base::boolalpha is not set, proceeds as with reading a long, except if the value is 1, sets *v* to true, if the value is 0, sets *v* to false, and otherwise set err to ios_base::failbit.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get< _CharT, _InIter >::do_get\(\)](#).

Referenced by [std::basic_istream< char >::operator>>\(\)](#), and [std::basic_istream< _CharT, _Traits >::operator>>\(\)](#).

get() [2/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    double & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

The input characters are parsed like the scanf g specifier. The matching type length modifier is also used.

The decimal point character used is numpunct::decimal_point(). Digit grouping is interpreted according to numpunct::grouping() and numpunct::thousands_sep(). If the pattern of digit groups isn't consistent, sets err to ios_base::failbit.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets err to ios_base::failbit and leaves *v* unaltered. Sets err to ios_base::eofbit if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get< _CharT, _InIter >::do_get\(\)](#).

get() [3/11]

```
template<typename _CharT, typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    float & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

The input characters are parsed like the `scanf g` specifier. The matching type length modifier is also used.

The decimal point character used is `num_punct::decimal_point()`. Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`. If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References `std::num_get< _CharT, _InIter >::do_get()`.

get() [4/11]

```
template<typename _CharT, typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used. Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
-------------------	------------------------

Parameters

<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get< _CharT, _InIter >::do_get\(\)](#).

get() [5/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long double & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable `v`. It does so by calling `num_get::do_get()`.

The input characters are parsed like the `scanf g` specifier. The matching type length modifier is also used.

The decimal point character used is `numpunct::decimal_point()`. Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get< _CharT, _InIter >::do_get\(\)](#).

get() [6/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    long long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used. Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References `std::num_get< _CharT, _InIter >::do_get()`.

get() [7/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned int & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used. Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get<_CharT, _InIter >::do_get\(\)](#).

get() [8/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used. Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered.

Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get<_CharT, _InIter >::do_get\(\)](#).

get() [9/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned long long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.
If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get<_CharT, _InIter >::do_get\(\)](#).

get() [10/11]

```
template<typename _CharT, typename _InIter >
iter_type std::num_get<_CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    unsigned short & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable `v`. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in `io`.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used. Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get<_CharT, _InIter >::do_get\(\)](#).

get() [11/11]

```
template<typename _CharT , typename _InIter >
iter_type std::num_get< _CharT, _InIter >::get (
    iter_type __in,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    void *& __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the pointer variable *v*. It does so by calling `num_get::do_get()`.

The input characters are parsed like the `scanf p` specifier.

Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

Note that the digit grouping effect for pointers is a bit ambiguous in the standard and shouldn't be relied on. See DR 344.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

References [std::num_get< _CharT, _InIter >::do_get\(\)](#).

6.749.5 Member Data Documentation**id**

```
template<typename _CharT , typename _InIter >
locale::id std::num_get< _CharT, _InIter >::id [static]
Numpunct facet id.
```

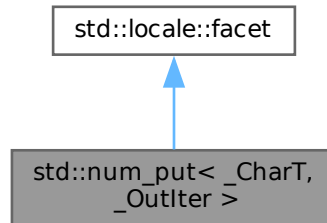
The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [locale_facets.tcc](#)

6.750 std::num_put< _CharT, _OutIter > Class Template Reference

```
#include <locale_facets.h>
```


Inheritance diagram for `std::num_put< _CharT, _Outlter >`:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Outlter` `iter_type`

Public Member Functions

- `num_put` (`size_t __refs=0`)
- template<typename `_ValueT` >
`_Outlter _M_insert_float` (`_Outlter __s`, `ios_base & __io`, `_CharT __fill`, `char __mod`, `_ValueT __v`) const
- template<typename `_ValueT` >
`_Outlter _M_insert_int` (`_Outlter __s`, `ios_base & __io`, `_CharT __fill`, `_ValueT __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `bool __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `const void *__v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `long __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `unsigned long __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `long long __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `unsigned long long __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `double __v`) const
- `iter_type put` (`iter_type __s`, `ios_base & __io`, `char_type __fill`, `long double __v`) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~num_put` ()
- void `_M_group_float` (`const char *__grouping`, `size_t __grouping_size`, `char_type __sep`, `const char_type *__p`, `char_type *__new`, `char_type *__cs`, `int & __len`) const
- void `_M_group_int` (`const char *__grouping`, `size_t __grouping_size`, `char_type __sep`, `ios_base & __io`, `char_type *__new`, `char_type *__cs`, `int & __len`) const

- template<typename _ValueT >
iter_type **_M_insert_float** (iter_type, ios_base & __io, char_type __fill, char __mod, _ValueT __v) const
- template<typename _ValueT >
iter_type **_M_insert_int** (iter_type, ios_base & __io, char_type __fill, _ValueT __v) const
- void **_M_pad** (char_type __fill, streamsize __w, ios_base & __io, char_type *__new, const char_type *__cs, int & __len) const
- virtual iter_type **do_put** (iter_type __s, ios_base & __io, char_type __fill, bool __v) const
- virtual iter_type **do_put** (iter_type __s, ios_base & __io, char_type __fill, long __v) const
- virtual iter_type **do_put** (iter_type __s, ios_base & __io, char_type __fill, unsigned long __v) const
- virtual iter_type **do_put** (iter_type __s, ios_base & __io, char_type __fill, long long __v) const
- virtual iter_type **do_put** (iter_type __s, ios_base & __io, char_type __fill, unsigned long long __v) const
- virtual iter_type **do_put** (iter_type, ios_base & __io, char_type, double) const
- virtual iter_type **do_put** (iter_type, ios_base & __io, char_type, long double) const
- virtual iter_type **do_put** (iter_type, ios_base & __io, char_type, const void *) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale & __cloc) throw ()
- static void **_S_create_c_locale** (__c_locale & __cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale & __cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_type_c_locale** (__c_locale __cloc, const char *__s)

6.750.1 Detailed Description

```
template<typename _CharT, typename _OutIter>
class std::num_put< _CharT, _OutIter >
```

Primary class template num_put.

This facet encapsulates the code to convert a number to a string. It is used by the ostream numeric insertion operators. The num_put template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the num_put facet.

6.750.2 Member Typedef Documentation

char_type

```
template<typename _CharT , typename _OutIter >
typedef _CharT std::num_put< _CharT, _OutIter >::char_type
```

Public typedefs.

iter_type

```
template<typename _CharT , typename _OutIter >
typedef _OutIter std::num_put< _CharT, _OutIter >::iter_type
```

Public typedefs.

6.750.3 Constructor & Destructor Documentation

num_put()

```
template<typename _CharT , typename _OutIter >
std::num_put< _CharT, _OutIter >::num_put (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~num_put()

```
template<typename _CharT , typename _OutIter >
virtual std::num_put< _CharT, _OutIter >::~~num_put ( ) [inline], [protected], [virtual]
```

Destructor.

6.750.4 Member Function Documentation

do_put() [1/8]

```
template<typename _CharT , typename _OutIter >
_OutIter std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    bool __v ) const [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::ios_base::M_getloc\(\)](#), [std::ios_base::adjustfield](#), [std::ios_base::boolalpha](#), [std::ios_base::flags\(\)](#), [std::ios_base::left](#), and [std::ios_base::width\(\)](#).

Referenced by [std::num_put< _CharT, _OutIter >::put\(\)](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), [std::num_put< _CharT, _OutIter >::put\(\)](#), and [std::num_put< _CharT, _OutIter >::put\(\)](#).

do_put() [2/8]

```
template<typename _CharT , typename _OutIter >
virtual iter_type std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long __v ) const [inline], [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

do_put() [3/8]

```
template<typename _CharT , typename _OutIter >
virtual iter_type std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long long __v ) const [inline], [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

do_put() [4/8]

```
template<typename _CharT , typename _OutIter >
virtual iter_type std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    unsigned long __v ) const [inline], [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

do_put() [5/8]

```
template<typename _CharT , typename _OutIter >
virtual iter_type std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    unsigned long long __v ) const [inline], [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

do_put() [6/8]

```
template<typename _CharT , typename _OutIter >
_OutIter std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const void * __v ) const [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::ios_base::basefield](#), [std::ios_base::flags\(\)](#), [std::ios_base::hex](#), [std::ios_base::showbase](#), and [std::ios_base::uppercase](#).

do_put() [7/8]

```
template<typename _CharT , typename _OutIter >
_OutIter std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    double __v ) const [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.

Parameters

<code>__v</code>	Value to format and insert.
------------------	-----------------------------

Returns

Iterator after writing.

do_put() [8/8]

```
template<typename _CharT , typename _OutIter >
_OutIter std::num_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long double __v ) const [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

put() [1/8]

```
template<typename _CharT , typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    bool __v ) const [inline]
```

Numeric formatting.

Formats the boolean `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

If `ios_base::boolalpha` is set, writes `ctype<CharT>::truename()` or `ctype<CharT>::falsename()`. Otherwise formats `v` as an int.

Parameters

<code>__s</code>	Stream to write to.
------------------	---------------------

Parameters

<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter >::do_put\(\)](#).

Referenced by [std::basic_ostream<_CharT, _Traits >::operator<<\(\)](#).

put() [2/8]

```
template<typename _CharT , typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const void * __v ) const [inline]
```

Numeric formatting.

Formats the pointer value `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

This function formats `v` as an unsigned long with `ios_base::hex` and `ios_base::showbase` set.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter >::do_put\(\)](#).

put() [3/8]

```
template<typename _CharT , typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    double __v ) const [inline]
```

Numeric formatting.

Formats the floating point value *v* and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of `io.flags()` & `ios_base::floatfield`. If equal to `ios_base::fixed`, formats like the `printf f` specifier. Else if equal to `ios_base::scientific`, formats like `e` or `E` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `g` or `G` depending on uppercase. Note that if both fixed and scientific are set, the effect will also be like `g` or `G`.

The output precision is given by `io.precision()`. This precision is capped at `numeric_limits::digits10 + 2` (different for double and long double). The default precision is 6.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showpoint` is set, a decimal point will always be output.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>↔ _s</code>	Stream to write to.
<code>↔ _io</code>	Source of locale and flags.
<code>↔ _fill</code>	Char_type to use for filling.
<code>↔ _v</code>	Value to format and insert.

Returns

Iterator after writing.

References `std::num_put< _CharT, _OutIter >::do_put()`.

put() [4/8]

```
template<typename _CharT , typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long __v ) const [inline]
```

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, formats like the `printf o` specifier. Else if equal to `ios_base::hex`, formats like `x` or `X` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `d`, `ld`, `lld` for signed and `u`, `lu`, `llu` for unsigned values. Note that if both oct and hex are set, neither will take effect.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showbase` is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

\leftarrow _s	Stream to write to.
\leftarrow _io	Source of locale and flags.
\leftarrow _fill	Char_type to use for filling.
\leftarrow _v	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter >::do_put\(\)](#).

put() [5/8]

```
template<typename _CharT , typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long double __v ) const [inline]
```

Numeric formatting.

Formats the floating point value *v* and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of `io.flags()` & `ios_base::floatfield`. If equal to `ios_base::fixed`, formats like the `printf f` specifier. Else if equal to `ios_base::scientific`, formats like `e` or `E` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `g` or `G` depending on uppercase. Note that if both fixed and scientific are set, the effect will also be like `g` or `G`.

The output precision is given by `io.precision()`. This precision is capped at `numeric_limits::digits10 + 2` (different for double and long double). The default precision is 6.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showpoint` is set, a decimal point will always be output.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

\leftarrow _s	Stream to write to.
\leftarrow _io	Source of locale and flags.
\leftarrow _fill	Char_type to use for filling.
\leftarrow _v	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter>::do_put\(\)](#).

put() [6/8]

```
template<typename _CharT, typename _OutIter>
iter_type std::num_put<_CharT, _OutIter>::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    long long __v ) const [inline]
```

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, formats like the `printf` `o` specifier. Else if equal to `ios_base::hex`, formats like `x` or `X` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `d`, `ld`, `lld` for signed and `u`, `lu`, `llu` for unsigned values. Note that if both `oct` and `hex` are set, neither will take effect.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showbase` is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter>::do_put\(\)](#).

put() [7/8]

```
template<typename _CharT, typename _OutIter>
iter_type std::num_put<_CharT, _OutIter>::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    unsigned long __v ) const [inline]
```

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling num_put::do_put().

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, formats like the printf o specifier. Else if equal to ios_base::hex, formats like x or X with ios_base::uppercase unset or set respectively. Otherwise, formats like d, ld, lld for signed and u, lu, llu for unsigned values. Note that if both oct and hex are set, neither will take effect.

If ios_base::showpos is set, '+' is output before positive values. If ios_base::showbase is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is numpunct::decimal_point(). Thousands separators are inserted according to numpunct::grouping() and numpunct::thousands_sep().

If io.width() is non-zero, enough *fill* characters are inserted to make the result at least that wide. If (io.flags() & ios_base::adjustfield) == ios_base::left, result is padded at the end. If ios_base::internal, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put< _CharT, _OutIter >::do_put\(\)](#).

put() [8/8]

```
template<typename _CharT, typename _OutIter >
iter_type std::num_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    unsigned long long __v ) const [inline]
```

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling num_put::do_put().

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, formats like the printf o specifier. Else if equal to ios_base::hex, formats like x or X with ios_base::uppercase unset or set respectively. Otherwise, formats like d, ld, lld for signed and u, lu, llu for unsigned values. Note that if both oct and hex are set, neither will take effect.

If ios_base::showpos is set, '+' is output before positive values. If ios_base::showbase is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is numpunct::decimal_point(). Thousands separators are inserted according to numpunct::grouping() and numpunct::thousands_sep().

If io.width() is non-zero, enough *fill* characters are inserted to make the result at least that wide. If (io.flags() & ios_base::adjustfield) == ios_base::left, result is padded at the end. If ios_base::internal, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>_↔_s</code>	Stream to write to.
<code>_↔_io</code>	Source of locale and flags.
<code>_↔_fill</code>	Char_type to use for filling.
<code>_↔_v</code>	Value to format and insert.

Returns

Iterator after writing.

References [std::num_put<_CharT, _OutIter>::do_put\(\)](#).

6.750.5 Member Data Documentation

id

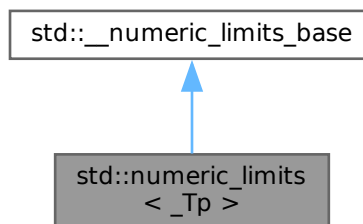
```
template<typename _CharT , typename _OutIter >
locale::id std::num_put< _CharT, _OutIter >::id [static]
Numpunct facet id.
```

The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [locale_facets.tcc](#)

6.751 std::numeric_limits<_Tp> Struct Template Reference

Inheritance diagram for std::numeric_limits<_Tp>:



Static Public Member Functions

- static constexpr `_Tp` [denorm_min](#) () noexcept
- static constexpr `_Tp` [epsilon](#) () noexcept
- static constexpr `_Tp` [infinity](#) () noexcept
- static constexpr `_Tp` [lowest](#) () noexcept

- static constexpr `_Tp` `max` () noexcept
- static constexpr `_Tp` `min` () noexcept
- static constexpr `_Tp` `quiet_NaN` () noexcept
- static constexpr `_Tp` `round_error` () noexcept
- static constexpr `_Tp` `signaling_NaN` () noexcept

Static Public Attributes

- static constexpr int `digits`
- static constexpr int `digits10`
- static constexpr `float_denorm_style` `has_denorm`
- static constexpr bool `has_denorm_loss`
- static constexpr bool `has_infinity`
- static constexpr bool `has_quiet_NaN`
- static constexpr bool `has_signaling_NaN`
- static constexpr bool `is_bounded`
- static constexpr bool `is_exact`
- static constexpr bool `is_iec559`
- static constexpr bool `is_integer`
- static constexpr bool `is_modulo`
- static constexpr bool `is_signed`
- static constexpr bool `is_specialized`
- static constexpr int `max_digits10`
- static constexpr int `max_exponent`
- static constexpr int `max_exponent10`
- static constexpr int `min_exponent`
- static constexpr int `min_exponent10`
- static constexpr int `radix`
- static constexpr `float_round_style` `round_style`
- static constexpr bool `tinyness_before`
- static constexpr bool `traps`

6.751.1 Detailed Description

template<typename `_Tp`>
struct `std::numeric_limits<_Tp>`

Properties of fundamental types.

This class allows a program to obtain information about the representation of a fundamental type on a given platform. For non-fundamental types, the functions will return 0 and the data members will all be `false`.

6.751.2 Member Function Documentation

`denorm_min()`

```
template<typename _Tp>
static constexpr _Tp std::numeric_limits<_Tp>::denorm_min ( ) [inline], [static], [constexpr],
[noexcept]
```

The minimum positive denormalized value. For types where `has_denorm` is `false`, this is the minimum positive normalized value.

epsilon()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::epsilon ( ) [inline], [static], [constexpr],
[noexcept]
```

The *machine epsilon*: the difference between 1 and the least value greater than 1 that is representable.

Referenced by [std::generate_canonical\(\)](#), [std::binomial_distribution< _IntType >::operator\(\)\(\)](#), [std::poisson_distribution< _IntType >::operator\(\)\(\)](#), and [std::operator<<\(\)](#).

infinity()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::infinity ( ) [inline], [static], [constexpr],
[noexcept]
```

The representation of positive infinity, if `has_infinity`.

lowest()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::lowest ( ) [inline], [static], [constexpr],
[noexcept]
```

A finite value `x` such that there is no other finite value `y` where `y < x`.

Referenced by [std::normal_distribution< _RealType >::min\(\)](#), [std::cauchy_distribution< _RealType >::min\(\)](#), [std::student_t_distribution< _RealType >::min\(\)](#), and [std::extreme_value_distribution< _RealType >::min\(\)](#).

max()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::max ( ) [inline], [static], [constexpr], [noexcept]
```

The maximum finite value.

Referenced by [std::normal_distribution< _RealType >::max\(\)](#), [std::lognormal_distribution< _RealType >::max\(\)](#), [std::gamma_distribution< _RealType >::max\(\)](#), [std::chi_squared_distribution< _RealType >::max\(\)](#), [std::cauchy_distribution< _RealType >::max\(\)](#), [std::fisher_f_distribution< _RealType >::max\(\)](#), [std::student_t_distribution< _RealType >::max\(\)](#), [std::bernoulli_distribution::max\(\)](#), [std::geometric_distribution< _IntType >::max\(\)](#), [std::negative_binomial_distribution< _IntType >::max\(\)](#), [std::poisson_distribution< _IntType >::max\(\)](#), [std::exponential_distribution< _RealType >::max\(\)](#), [std::weibull_distribution< _RealType >::max\(\)](#), [std::extreme_value_distribution< _RealType >::max\(\)](#), [std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::operator\(\)\(\)](#), [std::binomial_distribution< _IntType >::operator\(\)\(\)](#), [std::poisson_distribution< _IntType >::operator\(\)\(\)](#), and [std::operator<<\(\)](#).

min()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::min ( ) [inline], [static], [constexpr], [noexcept]
```

The minimum finite value, or for floating types with denormalization, the minimum positive normalized value.

Referenced by [std::bernoulli_distribution::min\(\)](#), and [std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::operator\(\)\(\)](#).

quiet_NaN()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::quiet_NaN ( ) [inline], [static], [constexpr],
[noexcept]
```

The representation of a quiet Not a Number, if `has_quiet_NaN`.

round_error()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::round_error ( ) [inline], [static], [constexpr],
[noexcept]
```

The maximum rounding error measurement (see LIA-1).

signaling_NaN()

```
template<typename _Tp >
static constexpr _Tp std::numeric_limits< _Tp >::signaling_NaN ( ) [inline], [static], [constexpr],
[noexcept]
```

The representation of a signaling Not a Number, if has_signaling_NaN.

6.751.3 Member Data Documentation**digits**

```
constexpr int std::__numeric_limits_base::digits [static], [constexpr], [inherited]
```

The number of radix digits that be represented without change: for integer types, the number of non-sign bits in the mantissa; for floating types, the number of radix digits in the mantissa.

digits10

```
constexpr int std::__numeric_limits_base::digits10 [static], [constexpr], [inherited]
```

The number of base 10 digits that can be represented without change.

has_denorm

```
constexpr float_denorm_style std::__numeric_limits_base::has_denorm [static], [constexpr], [inherited]
```

See std::float_denorm_style for more information.

has_denorm_loss

```
constexpr bool std::__numeric_limits_base::has_denorm_loss [static], [constexpr], [inherited]
```

True if loss of accuracy is detected as a denormalization loss, rather than as an inexact result.

has_infinity

```
constexpr bool std::__numeric_limits_base::has_infinity [static], [constexpr], [inherited]
```

True if the type has a representation for positive infinity.

has_quiet_NaN

```
constexpr bool std::__numeric_limits_base::has_quiet_NaN [static], [constexpr], [inherited]
```

True if the type has a representation for a quiet (non-signaling) Not a Number.

has_signaling_NaN

```
constexpr bool std::__numeric_limits_base::has_signaling_NaN [static], [constexpr], [inherited]
```

True if the type has a representation for a signaling Not a Number.

is_bounded

```
constexpr bool std::__numeric_limits_base::is_bounded [static], [constexpr], [inherited]
```

True if the set of values representable by the type is finite. All built-in types are bounded, this member would be false for arbitrary precision types.

is_exact

```
constexpr bool std::__numeric_limits_base::is_exact [static], [constexpr], [inherited]
```

True if the type uses an exact representation. All integer types are exact, but not all exact types are integer. For example, rational and fixed-exponent representations are exact but not integer.

is_iec559

```
constexpr bool std::__numeric_limits_base::is_iec559 [static], [constexpr], [inherited]
```

True if-and-only-if the type adheres to the IEC 559 standard, also known as IEEE 754. (Only makes sense for floating point types.)

is_integer

```
constexpr bool std::__numeric_limits_base::is_integer [static], [constexpr], [inherited]
```

True if the type is integer.

is_modulo

```
constexpr bool std::__numeric_limits_base::is_modulo [static], [constexpr], [inherited]
```

True if the type is *modulo*. A type is modulo if, for any operation involving +, -, or * on values of that type whose result would fall outside the range [min(),max()], the value returned differs from the true value by an integer multiple of max() - min() + 1. On most machines, this is false for floating types, true for unsigned integers, and true for signed integers. See PR22200 about signed integers.

is_signed

```
constexpr bool std::__numeric_limits_base::is_signed [static], [constexpr], [inherited]
```

True if the type is signed.

is_specialized

```
constexpr bool std::__numeric_limits_base::is_specialized [static], [constexpr], [inherited]
```

This will be true for all fundamental types (which have specializations), and false for everything else.

max_digits10

```
constexpr int std::__numeric_limits_base::max_digits10 [static], [constexpr], [inherited]
```

The number of base 10 digits required to ensure that values which differ are always differentiated.

max_exponent

```
constexpr int std::__numeric_limits_base::max_exponent [static], [constexpr], [inherited]
```

The maximum positive integer such that `radix` raised to the power of (one less than that integer) is a representable finite floating point number.

max_exponent10

```
constexpr int std::__numeric_limits_base::max_exponent10 [static], [constexpr], [inherited]
```

The maximum positive integer such that 10 raised to that power is in the range of representable finite floating point numbers.

min_exponent

```
constexpr int std::__numeric_limits_base::min_exponent [static], [constexpr], [inherited]
```

The minimum negative integer such that `radix` raised to the power of (one less than that integer) is a normalized floating point number.

min_exponent10

```
constexpr int std::__numeric_limits_base::min_exponent10 [static], [constexpr], [inherited]
```

The minimum negative integer such that 10 raised to that power is in the range of normalized floating point numbers.

radix

```
constexpr int std::__numeric_limits_base::radix [static], [constexpr], [inherited]
```

For integer types, specifies the base of the representation. For floating types, specifies the base of the exponent representation.

round_style

```
constexpr float\_round\_style std::__numeric_limits_base::round_style [static], [constexpr], [inherited]
```

See `std::float_round_style` for more information. This is only meaningful for floating types; integer types will all be `round_toward_zero`.

tinyness_before

```
constexpr bool std::__numeric_limits_base::tinyness_before [static], [constexpr], [inherited]
```

True if tininess is detected before rounding. (see IEC 559)

traps

```
constexpr bool std::__numeric_limits_base::traps [static], [constexpr], [inherited]
```

True if trapping is implemented for this type.

The documentation for this struct was generated from the following file:

- [limits](#)

6.752 `std::numeric_limits< bool >` Struct Reference

Static Public Member Functions

- static constexpr bool **denorm_min** () noexcept
- static constexpr bool **epsilon** () noexcept
- static constexpr bool **infinity** () noexcept
- static constexpr bool **lowest** () noexcept
- static constexpr bool **max** () noexcept
- static constexpr bool **min** () noexcept
- static constexpr bool **quiet_NaN** () noexcept
- static constexpr bool **round_error** () noexcept
- static constexpr bool **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.752.1 Detailed Description

`numeric_limits<bool>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.753 `std::numeric_limits< char >` Struct Reference

Static Public Member Functions

- static constexpr char **denorm_min** () noexcept
- static constexpr char **epsilon** () noexcept
- static constexpr char **infinity** () noexcept
- static constexpr char **lowest** () noexcept
- static constexpr char **max** () noexcept
- static constexpr char **min** () noexcept
- static constexpr char **quiet_NaN** () noexcept
- static constexpr char **round_error** () noexcept
- static constexpr char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.753.1 Detailed Description

`numeric_limits<char>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.754 `std::numeric_limits< char16_t >` Struct Reference

Static Public Member Functions

- static constexpr `char16_t` **denorm_min** () noexcept
- static constexpr `char16_t` **epsilon** () noexcept
- static constexpr `char16_t` **infinity** () noexcept
- static constexpr `char16_t` **lowest** () noexcept
- static constexpr `char16_t` **max** () noexcept
- static constexpr `char16_t` **min** () noexcept
- static constexpr `char16_t` **quiet_NaN** () noexcept
- static constexpr `char16_t` **round_error** () noexcept
- static constexpr `char16_t` **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr `float_denorm_style` **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr `float_round_style` **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.754.1 Detailed Description

`numeric_limits<char16_t>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.755 std::numeric_limits< char32_t > Struct Reference

Static Public Member Functions

- static constexpr char32_t **denorm_min** () noexcept
- static constexpr char32_t **epsilon** () noexcept
- static constexpr char32_t **infinity** () noexcept
- static constexpr char32_t **lowest** () noexcept
- static constexpr char32_t **max** () noexcept
- static constexpr char32_t **min** () noexcept
- static constexpr char32_t **quiet_NaN** () noexcept
- static constexpr char32_t **round_error** () noexcept
- static constexpr char32_t **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr float_denorm_style **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr float_round_style **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.755.1 Detailed Description

numeric_limits<char32_t> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.756 `std::numeric_limits< double >` Struct Reference

Static Public Member Functions

- static constexpr double **denorm_min** () noexcept
- static constexpr double **epsilon** () noexcept
- static constexpr double **infinity** () noexcept
- static constexpr double **lowest** () noexcept
- static constexpr double **max** () noexcept
- static constexpr double **min** () noexcept
- static constexpr double **quiet_NaN** () noexcept
- static constexpr double **round_error** () noexcept
- static constexpr double **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.756.1 Detailed Description

`numeric_limits<double>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.757 std::numeric_limits< float > Struct Reference

Static Public Member Functions

- static constexpr float **denorm_min** () noexcept
- static constexpr float **epsilon** () noexcept
- static constexpr float **infinity** () noexcept
- static constexpr float **lowest** () noexcept
- static constexpr float **max** () noexcept
- static constexpr float **min** () noexcept
- static constexpr float **quiet_NaN** () noexcept
- static constexpr float **round_error** () noexcept
- static constexpr float **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.757.1 Detailed Description

numeric_limits<float> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.758 `std::numeric_limits< int >` Struct Reference

Static Public Member Functions

- static constexpr int **denorm_min** () noexcept
- static constexpr int **epsilon** () noexcept
- static constexpr int **infinity** () noexcept
- static constexpr int **lowest** () noexcept
- static constexpr int **max** () noexcept
- static constexpr int **min** () noexcept
- static constexpr int **quiet_NaN** () noexcept
- static constexpr int **round_error** () noexcept
- static constexpr int **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.758.1 Detailed Description

`numeric_limits<int>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.759 std::numeric_limits< long > Struct Reference

Static Public Member Functions

- static constexpr long **denorm_min** () noexcept
- static constexpr long **epsilon** () noexcept
- static constexpr long **infinity** () noexcept
- static constexpr long **lowest** () noexcept
- static constexpr long **max** () noexcept
- static constexpr long **min** () noexcept
- static constexpr long **quiet_NaN** () noexcept
- static constexpr long **round_error** () noexcept
- static constexpr long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.759.1 Detailed Description

numeric_limits<long> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.760 `std::numeric_limits< long double >` Struct Reference

Static Public Member Functions

- static constexpr long double **denorm_min** () noexcept
- static constexpr long double **epsilon** () noexcept
- static constexpr long double **infinity** () noexcept
- static constexpr long double **lowest** () noexcept
- static constexpr long double **max** () noexcept
- static constexpr long double **min** () noexcept
- static constexpr long double **quiet_NaN** () noexcept
- static constexpr long double **round_error** () noexcept
- static constexpr long double **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.760.1 Detailed Description

`numeric_limits<long double>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.761 std::numeric_limits< long long > Struct Reference

Static Public Member Functions

- static constexpr long long **denorm_min** () noexcept
- static constexpr long long **epsilon** () noexcept
- static constexpr long long **infinity** () noexcept
- static constexpr long long **lowest** () noexcept
- static constexpr long long **max** () noexcept
- static constexpr long long **min** () noexcept
- static constexpr long long **quiet_NaN** () noexcept
- static constexpr long long **round_error** () noexcept
- static constexpr long long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr float_denorm_style **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr float_round_style **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.761.1 Detailed Description

numeric_limits<long long> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.762 `std::numeric_limits< short >` Struct Reference

Static Public Member Functions

- static constexpr short **denorm_min** () noexcept
- static constexpr short **epsilon** () noexcept
- static constexpr short **infinity** () noexcept
- static constexpr short **lowest** () noexcept
- static constexpr short **max** () noexcept
- static constexpr short **min** () noexcept
- static constexpr short **quiet_NaN** () noexcept
- static constexpr short **round_error** () noexcept
- static constexpr short **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.762.1 Detailed Description

`numeric_limits<short>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.763 std::numeric_limits< signed char > Struct Reference

Static Public Member Functions

- static constexpr signed char **denorm_min** () noexcept
- static constexpr signed char **epsilon** () noexcept
- static constexpr signed char **infinity** () noexcept
- static constexpr signed char **lowest** () noexcept
- static constexpr signed char **max** () noexcept
- static constexpr signed char **min** () noexcept
- static constexpr signed char **quiet_NaN** () noexcept
- static constexpr signed char **round_error** () noexcept
- static constexpr signed char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.763.1 Detailed Description

numeric_limits<signed char> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.764 `std::numeric_limits< unsigned char >` Struct Reference

Static Public Member Functions

- static constexpr unsigned char **denorm_min** () noexcept
- static constexpr unsigned char **epsilon** () noexcept
- static constexpr unsigned char **infinity** () noexcept
- static constexpr unsigned char **lowest** () noexcept
- static constexpr unsigned char **max** () noexcept
- static constexpr unsigned char **min** () noexcept
- static constexpr unsigned char **quiet_NaN** () noexcept
- static constexpr unsigned char **round_error** () noexcept
- static constexpr unsigned char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.764.1 Detailed Description

`numeric_limits<unsigned char>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.765 std::numeric_limits< unsigned int > Struct Reference

Static Public Member Functions

- static constexpr unsigned int **denorm_min** () noexcept
- static constexpr unsigned int **epsilon** () noexcept
- static constexpr unsigned int **infinity** () noexcept
- static constexpr unsigned int **lowest** () noexcept
- static constexpr unsigned int **max** () noexcept
- static constexpr unsigned int **min** () noexcept
- static constexpr unsigned int **quiet_NaN** () noexcept
- static constexpr unsigned int **round_error** () noexcept
- static constexpr unsigned int **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.765.1 Detailed Description

numeric_limits<unsigned int> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.766 `std::numeric_limits< unsigned long >` Struct Reference

Static Public Member Functions

- static constexpr unsigned long **denorm_min** () noexcept
- static constexpr unsigned long **epsilon** () noexcept
- static constexpr unsigned long **infinity** () noexcept
- static constexpr unsigned long **lowest** () noexcept
- static constexpr unsigned long **max** () noexcept
- static constexpr unsigned long **min** () noexcept
- static constexpr unsigned long **quiet_NaN** () noexcept
- static constexpr unsigned long **round_error** () noexcept
- static constexpr unsigned long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.766.1 Detailed Description

`numeric_limits<unsigned long>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.767 std::numeric_limits< unsigned long long > Struct Reference

Static Public Member Functions

- static constexpr unsigned long long **denorm_min** () noexcept
- static constexpr unsigned long long **epsilon** () noexcept
- static constexpr unsigned long long **infinity** () noexcept
- static constexpr unsigned long long **lowest** () noexcept
- static constexpr unsigned long long **max** () noexcept
- static constexpr unsigned long long **min** () noexcept
- static constexpr unsigned long long **quiet_NaN** () noexcept
- static constexpr unsigned long long **round_error** () noexcept
- static constexpr unsigned long long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.767.1 Detailed Description

numeric_limits<unsigned long long> specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.768 `std::numeric_limits< unsigned short >` Struct Reference

Static Public Member Functions

- static constexpr unsigned short **denorm_min** () noexcept
- static constexpr unsigned short **epsilon** () noexcept
- static constexpr unsigned short **infinity** () noexcept
- static constexpr unsigned short **lowest** () noexcept
- static constexpr unsigned short **max** () noexcept
- static constexpr unsigned short **min** () noexcept
- static constexpr unsigned short **quiet_NaN** () noexcept
- static constexpr unsigned short **round_error** () noexcept
- static constexpr unsigned short **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.768.1 Detailed Description

`numeric_limits<unsigned short>` specialization.

The documentation for this struct was generated from the following file:

- [limits](#)

6.769 std::numeric_limits< wchar_t > Struct Reference

Static Public Member Functions

- static constexpr wchar_t **denorm_min** () noexcept
- static constexpr wchar_t **epsilon** () noexcept
- static constexpr wchar_t **infinity** () noexcept
- static constexpr wchar_t **lowest** () noexcept
- static constexpr wchar_t **max** () noexcept
- static constexpr wchar_t **min** () noexcept
- static constexpr wchar_t **quiet_NaN** () noexcept
- static constexpr wchar_t **round_error** () noexcept
- static constexpr wchar_t **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr float_denorm_style **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr float_round_style **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

6.769.1 Detailed Description

numeric_limits<wchar_t> specialization.

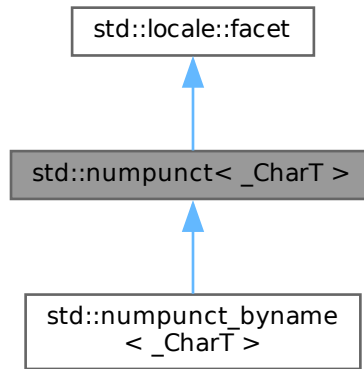
The documentation for this struct was generated from the following file:

- [limits](#)

6.770 std::num_punct<_CharT> Class Template Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::num_punct<_CharT>:



Public Types

- typedef `__num_punct_cache<_CharT>` `__cache_type`
- typedef `_CharT` `char_type`
- typedef `basic_string<_CharT>` `string_type`

Public Member Functions

- `num_punct` (`__c_locale __cloc`, `size_t __refs=0`)
- `num_punct` (`__cache_type *__cache`, `size_t __refs=0`)
- `num_punct` (`size_t __refs=0`)
- `char_type decimal_point` () const
- `string_type falsename` () const
- `string_type grouping` () const
- `char_type thousands_sep` () const
- `string_type truename` () const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~num_punct` ()
- void `_M_initialize_num_punct` (`__c_locale __cloc`)
- void `_M_initialize_num_punct` (`__c_locale __cloc`)
- void `_M_initialize_num_punct` (`__c_locale __cloc=0`)
- virtual `char_type do_decimal_point` () const

- virtual [string_type do_falsename](#) () const
- virtual [string do_grouping](#) () const
- virtual [char_type do_thousands_sep](#) () const
- virtual [string_type do_truename](#) () const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_ctype_c_locale (__c_locale __cloc, const char *__s)`

Protected Attributes

- `__cache_type * _M_data`

6.770.1 Detailed Description

template<typename _CharT>
class std::num_punct<_CharT>

Primary class template num_punct.

This facet stores several pieces of information related to printing and scanning numbers, such as the decimal point character. It takes a template parameter specifying the char type. The num_punct facet is used by streams for many I/O operations involving numbers.

The num_punct template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from a num_punct facet.

6.770.2 Member Typedef Documentation

char_type

```
template<typename _CharT>
typedef _CharT std::num_punct<_CharT>::char_type
```

Public typedefs.

string_type

```
template<typename _CharT>
typedef basic_string<_CharT> std::num_punct<_CharT>::string_type
```

Public typedefs.

6.770.3 Constructor & Destructor Documentation

num_punct() [1/3]

```
template<typename _CharT>
std::num_punct<_CharT>::num_punct (
    size_t __refs = 0 ) [inline], [explicit]
```

Num_punct constructor.

do_decimal_point()

```
template<typename _CharT >
virtual char_type std::num_punct<_CharT>::do_decimal_point ( ) const [inline], [protected], [virtual]
```

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Referenced by `std::num_punct<_CharT>::decimal_point()`.

do_falsename()

```
template<typename _CharT >
virtual string_type std::num_punct<_CharT>::do_falsename ( ) const [inline], [protected], [virtual]
```

Return string representation of bool false.

Returns a `string_type` containing the text representation for false bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of false.

Referenced by `std::num_punct<_CharT>::falsename()`.

do_grouping()

```
template<typename _CharT >
virtual string std::num_punct<_CharT>::do_grouping ( ) const [inline], [protected], [virtual]
```

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Referenced by `std::num_punct<_CharT>::grouping()`.

do_thousands_sep()

```
template<typename _CharT >
virtual char_type std::num_punct<_CharT>::do_thousands_sep ( ) const [inline], [protected], [virtual]
```

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Referenced by `std::num_punct<_CharT>::thousands_sep()`.

do_truename()

```
template<typename _CharT >
```

```
virtual string_type std::num_punct< _CharT >::do_truename ( ) const [inline], [protected], [virtual]
```

Return string representation of bool true.

Returns a string_type containing the text representation for true bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of true.

Referenced by [std::num_punct< _CharT >::truename\(\)](#).

falsename()

```
template<typename _CharT >
```

```
string_type std::num_punct< _CharT >::false_name ( ) const [inline]
```

Return string representation of bool false.

This function returns a string_type containing the text representation for false bool variables. It does so by calling num_punct<char_type>::do_false_name().

Returns

string_type representing printed form of false.

References [std::num_punct< _CharT >::do_false_name\(\)](#).

grouping()

```
template<typename _CharT >
```

```
string std::num_punct< _CharT >::grouping ( ) const [inline]
```

Return grouping specification.

This function returns a string representing groupings for the integer part of a number. Groupings indicate where thousands separators should be inserted in the integer part of a number.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the grouping() returns "\003\002" and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was "32", this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling num_punct<char_type>::do_grouping().

Returns

string representing grouping specification.

References [std::num_punct< _CharT >::do_grouping\(\)](#).

thousands_sep()

```
template<typename _CharT >
```

```
char_type std::num_punct< _CharT >::thousands_sep ( ) const [inline]
```

Return thousands separator character.

This function returns a char_type to use as a thousands separator. It does so by returning num_punct<char_type>::do_thousands_sep().

Returns

char_type representing a thousands separator.

References [std::num_punct< _CharT >::do_thousands_sep\(\)](#).

truename()

```
template<typename _CharT >
string_type std::numpunct<_CharT>::true_name ( ) const [inline]
```

Return string representation of bool true.

This function returns a string_type containing the text representation for true bool variables. It does so by calling numpunct<char_type>::do_true_name().

Returns

string_type representing printed form of true.

References [std::numpunct<_CharT>::do_true_name\(\)](#).

6.770.5 Member Data Documentation**id**

```
template<typename _CharT >
locale::id std::numpunct<_CharT>::id [static]
```

Numpunct facet id.

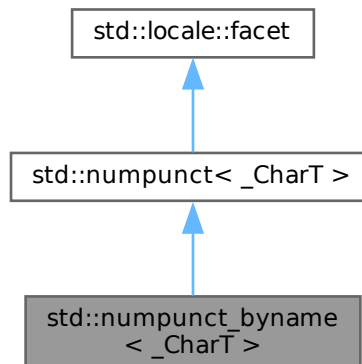
The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.771 std::numpunct_byname<_CharT> Class Template Reference

```
#include <locale_facets.h>
```

Inheritance diagram for std::numpunct_byname<_CharT>:

**Public Types**

- typedef __numpunct_cache<_CharT> **__cache_type**
- typedef _CharT **char_type**
- typedef [basic_string](#)<_CharT> **string_type**

Public Member Functions

- **numpunct_byname** (const char *__s, size_t __refs=0)
- **numpunct_byname** (const [string](#) & __s, size_t __refs=0)
- char_type **decimal_point** () const
- [string_type](#) **falsename** () const
- [string](#) **grouping** () const
- char_type **thousands_sep** () const
- [string_type](#) **truename** () const

Static Public Attributes

- static [locale::id](#) **id**

Protected Member Functions

- void **_M_initialize_numpunct** (__c_locale __cloc)
- void **_M_initialize_numpunct** (__c_locale __cloc)
- void **_M_initialize_numpunct** (__c_locale __cloc=0)
- virtual char_type **do_decimal_point** () const
- virtual [string_type](#) **do_falsename** () const
- virtual [string](#) **do_grouping** () const
- virtual char_type **do_thousands_sep** () const
- virtual [string_type](#) **do_truename** () const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale & __cloc) throw ()
- static void **_S_create_c_locale** (__c_locale & __cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale & __cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_type_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __cache_type * **_M_data**

6.771.1 Detailed Description

```
template<typename _CharT>
class std::numpunct_byname< _CharT >
```

class numpunct_byname [22.2.3.2].

6.771.2 Member Function Documentation

decimal_point()

```
template<typename _CharT >
char_type std::numpunct< _CharT >::decimal_point ( ) const [inline], [inherited]
```

Return decimal point character.

This function returns a char_type to use as a decimal point. It does so by returning `numpunct<char_type>::do_decimal_point()`.

Returns

char_type representing a decimal point.

References `std::num_punct<_CharT>::do_decimal_point()`.

do_decimal_point()

```
template<typename _CharT >
virtual char_type std::num_punct<_CharT>::do_decimal_point ( ) const [inline], [protected],
[virtual], [inherited]
```

Return decimal point character.

Returns a *char_type* to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Referenced by `std::num_punct<_CharT>::decimal_point()`.

do_falsename()

```
template<typename _CharT >
virtual string_type std::num_punct<_CharT>::do_falsename ( ) const [inline], [protected], [virtual],
[inherited]
```

Return string representation of bool false.

Returns a *string_type* containing the text representation for false bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of false.

Referenced by `std::num_punct<_CharT>::falsename()`.

do_grouping()

```
template<typename _CharT >
virtual string std::num_punct<_CharT>::do_grouping ( ) const [inline], [protected], [virtual],
[inherited]
```

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Referenced by `std::num_punct<_CharT>::grouping()`.

do_thousands_sep()

```
template<typename _CharT >
virtual char_type std::num_punct<_CharT>::do_thousands_sep ( ) const [inline], [protected],
[virtual], [inherited]
```

Return thousands separator character.

Returns a *char_type* to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Referenced by [std::num_punct<_CharT>::thousands_sep\(\)](#).

do_truename()

```
template<typename _CharT >
virtual string_type std::num_punct<_CharT>::do_truename ( ) const [inline], [protected], [virtual],
[inherited]
```

Return string representation of bool true.

Returns a *string_type* containing the text representation for true bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of true.

Referenced by [std::num_punct<_CharT>::truename\(\)](#).

falsename()

```
template<typename _CharT >
string_type std::num_punct<_CharT>::false_name ( ) const [inline], [inherited]
```

Return string representation of bool false.

This function returns a *string_type* containing the text representation for false bool variables. It does so by calling `num_punct<char_type>::do_false_name()`.

Returns

string_type representing printed form of false.

References [std::num_punct<_CharT>::do_false_name\(\)](#).

grouping()

```
template<typename _CharT >
string std::num_punct<_CharT>::grouping ( ) const [inline], [inherited]
```

Return grouping specification.

This function returns a string representing groupings for the integer part of a number. Groupings indicate where thousands separators should be inserted in the integer part of a number.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the `grouping()` returns `"\003\002"` and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was `"32"`, this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling `num_punct<char_type>::do_grouping()`.

Returns

string representing grouping specification.

References [std::num_punct<_CharT>::do_grouping\(\)](#).

thousands_sep()

```
template<typename _CharT >
```

```
char_type std::numpunct< _CharT >::thousands_sep ( ) const [inline], [inherited]
```

Return thousands separator character.

This function returns a char_type to use as a thousands separator. It does so by returning returning numpunct<char↵_type>::do_thousands_sep().

Returns

char_type representing a thousands separator.

References [std::numpunct< _CharT >::do_thousands_sep\(\)](#).

truename()

```
template<typename _CharT >
```

```
string_type std::numpunct< _CharT >::trueName ( ) const [inline], [inherited]
```

Return string representation of bool true.

This function returns a string_type containing the text representation for true bool variables. It does so by calling numpunct<char_type>::do_trueName().

Returns

string_type representing printed form of true.

References [std::numpunct< _CharT >::do_trueName\(\)](#).

6.771.3 Member Data Documentation**id**

```
template<typename _CharT >
```

```
locale::id std::numpunct< _CharT >::id [static], [inherited]
```

Numpunct facet id.

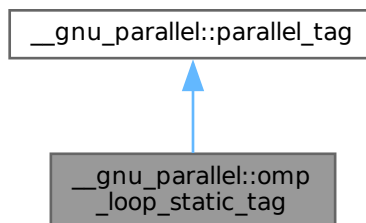
The documentation for this class was generated from the following file:

- [locale_facets.h](#)

6.772 __gnu_parallel::omp_loop_static_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::omp_loop_static_tag:



Public Member Functions

- [_ThreadIndex __get_num_threads \(\)](#)
- void [set_num_threads \(_ThreadIndex __num_threads\)](#)

6.772.1 Detailed Description

Recommends parallel execution using OpenMP static load-balancing at compile time.

6.772.2 Member Function Documentation

[__get_num_threads\(\)](#)

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#).

[set_num_threads\(\)](#)

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

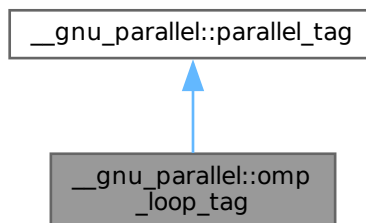
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.773 __gnu_parallel::omp_loop_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::omp_loop_tag`:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

6.773.1 Detailed Description

Recommends parallel execution using OpenMP dynamic load-balancing at compile time.

6.773.2 Member Function Documentation**__get_num_threads()**

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#).

set_num_threads()

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.774 std::once_flag Struct Reference**Public Member Functions**

- [once_flag](#) (const [once_flag](#) &)=delete
- [once_flag](#) & [operator=](#) (const [once_flag](#) &)=delete

Friends

- [template<typename _Callable, typename... _Args>](#)
void [call_once](#) ([once_flag](#) &__once, _Callable &&__f, _Args &&... __args)

6.774.1 Detailed Description

Flag type used by std::call_once.

6.774.2 Constructor & Destructor Documentation**once_flag()**

```
std::once_flag::once_flag (
    const once\_flag & ) [delete]
```


Deleted copy constructor.

6.774.3 Member Function Documentation

operator=()

```
once_flag & std::once_flag::operator= (
    const once_flag & ) [delete]
```

Deleted assignment operator.

6.774.4 Friends And Related Symbol Documentation

call_once

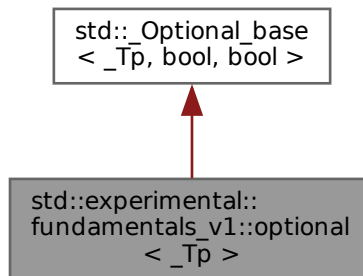
```
template<typename _Callable , typename... _Args>
void call_once (
    once_flag & __once,
    _Callable && __f,
    _Args &&... __args ) [friend]
```

Invoke a callable and synchronize with other calls using the same flag.
The documentation for this struct was generated from the following file:

- [mutex](#)

6.775 std::experimental::fundamentals_v1::optional<_Tp> Class Template Reference

Inheritance diagram for std::experimental::fundamentals_v1::optional<_Tp>:



Public Types

- using **value_type** = _Tp

Public Member Functions

- template<typename _Up = _Tp, [enable_if_t](#)< __and_< __not_< [is_same](#)< optional<_Tp>, [decay_t](#)<_Up>>>, [is_constructible](#)<_Tp, _Up &&>, [is_convertible](#)<_Up &&, _Tp>>::value, bool> = true>
constexpr **optional** (_Up &&__t)

- template<typename _Up = _Tp, [enable_if_t](#)< __and_< __not_< [is_same](#)< [optional](#)< _Tp >, [decay_t](#)< _Up >>, [is_constructible](#)< _Tp, _Up && >, __not_< [is_convertible](#)< _Up &&, _Tp >>>::value, bool > = false>
constexpr **optional** (_Up &&__t)
- template<typename _Up, [enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, const _Up & >, [is_convertible](#)< const _Up &, _Tp >, __not_< __converts_from_optional< _Tp, _Up >>>::value, bool > = true>
constexpr **optional** (const [optional](#)< _Up > &__t)
- template<typename _Up, [enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, const _Up & >, __not_< [is_convertible](#)< const _Up &, _Tp >>, __not_< __converts_from_optional< _Tp, _Up >>>::value, bool > = false>
constexpr **optional** (const [optional](#)< _Up > &__t)
- template<typename _Up, [enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, _Up && >, [is_convertible](#)< _Up &&, _Tp >, __not_< __converts_from_optional< _Tp, _Up >>>::value, bool > = true>
constexpr **optional** ([optional](#)< _Up > &&__t)
- template<typename _Up, [enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, _Up && >, __not_< __converts_from_optional< _Up &&, _Tp >>>::value, bool > = false>
constexpr **optional** ([optional](#)< _Up > &&__t)
- template<typename... _Args>
[enable_if_t](#)< [is_constructible](#)< _Tp, _Args &&... >::value > **emplace** (_Args &&... __args)
- template<typename _Up, typename... _Args>
[enable_if_t](#)< [is_constructible](#)< _Tp, [initializer_list](#)< _Up > &, _Args &&... >::value > **emplace** ([initializer_list](#)< _Up > __il, _Args &&... __args)
- constexpr **operator bool** () const noexcept
- constexpr _Tp & **operator*** () &
- constexpr _Tp && **operator*** () &&
- constexpr const _Tp & **operator*** () const &
- constexpr const _Tp && **operator*** () const &&
- _Tp * **operator->** ()
- constexpr const _Tp * **operator->** () const
- template<typename _Up = _Tp>
[enable_if_t](#)< __and_< __not_< [is_same](#)< [optional](#)< _Tp >, [decay_t](#)< _Up >>, [is_constructible](#)< _Tp, _Up >, __not_< __and_< [is_scalar](#)< _Tp >, [is_same](#)< _Tp, [decay_t](#)< _Up >>>, [is_assignable](#)< _Tp &, _Up >>>::value, [optional](#) & > **operator=** (_Up &&__u)
- template<typename _Up >
[enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, const _Up & >, [is_assignable](#)< _Tp &, _Up >, __not_< __converts_from_optional< _Tp, _Up >>, __not_< __assigns_from_optional< _Tp, _Up >>>::value, [optional](#) & > **operator=** (const [optional](#)< _Up > &__u)
- [optional](#) & **operator=** ([nullopt_t](#)) noexcept
- template<typename _Up >
[enable_if_t](#)< __and_< __not_< [is_same](#)< _Tp, _Up >, [is_constructible](#)< _Tp, _Up >, [is_assignable](#)< _Tp &, _Up >, __not_< __converts_from_optional< _Tp, _Up >>, __not_< __assigns_from_optional< _Tp, _Up >>>::value, [optional](#) & > **operator=** ([optional](#)< _Up > &&__u)
- void **swap** ([optional](#) &__other) noexcept([is_nothrow_move_constructible](#)< _Tp >()) &&__is_nothrow_swappable< _Tp >::value)
- constexpr _Tp & **value** () &
- constexpr _Tp && **value** () &&
- constexpr const _Tp & **value** () const &
- constexpr const _Tp && **value** () const &&
- template<typename _Up >
_Tp **value_or** (_Up &&__u) &&
- template<typename _Up >
constexpr _Tp **value_or** (_Up &&__u) const &

6.775.1 Detailed Description

```
template<typename _Tp>
class std::experimental::fundamentals_v1::optional< _Tp >
```

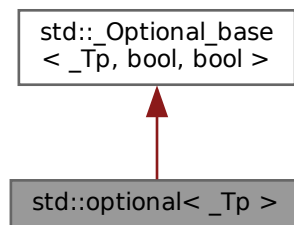
Class template for optional values.

The documentation for this class was generated from the following file:

- [experimental/optional](#)

6.776 std::optional< _Tp > Class Template Reference

Inheritance diagram for std::optional< _Tp >:



Public Types

- using **value_type** = _Tp

Public Member Functions

- template<typename _Up = _Tp, _Requires< __not_self< _Up >, __not_tag< _Up >, [is_constructible](#)< _Tp, _Up >, [is_convertible](#)< _Up, _Tp > > = true>

constexpr **optional** (_Up &&__t) noexcept(is_nothrow_constructible_v< _Tp, _Up >)
- template<typename _Up = _Tp, _Requires< __not_self< _Up >, __not_tag< _Up >, [is_constructible](#)< _Tp, _Up >, __not_< [is_convertible](#)< _Up, _Tp > > > = false>

constexpr **optional** (_Up &&__t) noexcept(is_nothrow_constructible_v< _Tp, _Up >)
- template<typename _Up, _Requires< __not_< [is_same](#)< _Tp, _Up > >, [is_constructible](#)< _Tp, const _Up &, [is_convertible](#)< const _Up &, _Tp >, __not_< __converts_from_optional< _Tp, _Up > > > = true>

constexpr **optional** (const [optional](#)< _Up > &&__t) noexcept(is_nothrow_constructible_v< _Tp, const _Up & >)
- template<typename _Up, _Requires< __not_< [is_same](#)< _Tp, _Up > >, [is_constructible](#)< _Tp, const _Up &, __not_< [is_convertible](#)< const _Up &, _Tp > >, __not_< __converts_from_optional< _Tp, _Up > > > = false>

constexpr **optional** (const [optional](#)< _Up > &&__t) noexcept(is_nothrow_constructible_v< _Tp, const _Up & >)
- template<typename... _Args, _Requires< [is_constructible](#)< _Tp, _Args... > > = false>

constexpr **optional** (in_place_t, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, _Args... >)
- template<typename _Up, typename... _Args, _Requires< [is_constructible](#)< _Tp, [initializer_list](#)< _Up > &, _Args... > > = false>

constexpr **optional** (in_place_t, [initializer_list](#)< _Up > __il, _Args &&... __args) noexcept(is_nothrow_<__constructible_v< _Tp, [initializer_list](#)< _Up > &, _Args... >)
- constexpr **optional** (nullopt_t) noexcept

- template<typename _Up, _Requires<__not_< is_same<_Tp, _Up>>, is_constructible<_Tp, _Up>, is_convertible<_Up, _Tp>, __not_<__converts_from_optional<_Tp, _Up>>> = true>
constexpr optional (optional<_Up> &&__t) noexcept(is_nothrow_constructible_v<_Tp, _Up>)
- template<typename _Up, _Requires<__not_< is_same<_Tp, _Up>>, is_constructible<_Tp, _Up>, __not_< is_convertible<_Up, _Tp>>, __not_<__converts_from_optional<_Tp, _Up>>> = false>
constexpr optional (optional<_Up> &&__t) noexcept(is_nothrow_constructible_v<_Tp, _Up>)
- template<typename... _Args>
constexpr enable_if_t< is_constructible_v<_Tp, _Args...>, _Tp &> emplace (_Args &&... __args)
noexcept(is_nothrow_constructible_v<_Tp, _Args...>)
- template<typename _Up, typename... _Args>
constexpr enable_if_t< is_constructible_v<_Tp, initializer_list<_Up> &, _Args...>, _Tp &> emplace
(initializer_list<_Up> __il, _Args &&... __args) noexcept(is_nothrow_constructible_v<_Tp, initializer_list<_Up> &, _Args...>)
- constexpr bool has_value () const noexcept
- constexpr operator bool () const noexcept
- constexpr _Tp && operator* () &&noexcept
- constexpr _Tp & operator* () &noexcept
- constexpr const _Tp && operator* () const &&noexcept
- constexpr const _Tp & operator* () const &noexcept
- constexpr const _Tp * operator-> () const noexcept
- constexpr _Tp * operator-> () noexcept
- template<typename _Up = _Tp>
constexpr enable_if_t<__and_v<__not_self<_Up>, __not_<__and_< is_scalar<_Tp>, is_same<_Tp, decay_t<_Up>>>>, is_constructible<_Tp, _Up>, is_assignable<_Tp &, _Up>>, optional &>
operator= (_Up &&__u) noexcept(__and_v< is_nothrow_constructible<_Tp, _Up>, is_nothrow_assignable<_Tp &, _Up>>)
- template<typename _Up>
constexpr enable_if_t<__and_v<__not_< is_same<_Tp, _Up>>, is_constructible<_Tp, const _Up &>, is_assignable<_Tp &, const _Up &>, __not_<__converts_from_optional<_Tp, _Up>>, __not_<__assigns_from_optional<_Tp, _Up>>>, optional &> operator= (const optional<_Up> &&__u) noexcept(__and_v< is_nothrow_constructible<_Tp, const _Up &>, is_nothrow_assignable<_Tp &, const _Up &>>)
- constexpr optional & operator= (nullopt_t) noexcept
- template<typename _Up>
constexpr enable_if_t<__and_v<__not_< is_same<_Tp, _Up>>, is_constructible<_Tp, _Up>, is_assignable<_Tp &, _Up>, __not_<__converts_from_optional<_Tp, _Up>>, __not_<__assigns_from_optional<_Tp, _Up>>>, optional &> operator= (optional<_Up> &&__u) noexcept(__and_v< is_nothrow_constructible<_Tp, _Up>, is_nothrow_assignable<_Tp &, _Up>>)
- constexpr void reset () noexcept
- constexpr void swap (optional &__other) noexcept(is_nothrow_move_constructible_v<_Tp> && is_nothrow_swappable_v<_Tp>)
- constexpr _Tp & value () &
- constexpr _Tp && value () &&
- constexpr const _Tp & value () const &
- constexpr const _Tp && value () const &&
- template<typename _Up>
constexpr _Tp value_or (_Up &&__u) &&
- template<typename _Up>
constexpr _Tp value_or (_Up &&__u) const &

6.776.1 Detailed Description

```
template<typename _Tp>
class std::optional< _Tp >
```

Class template for optional values.

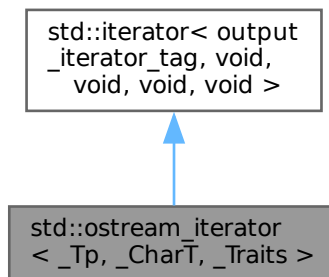
The documentation for this class was generated from the following file:

- [optional](#)

6.777 std::ostream_iterator< _Tp, _CharT, _Traits > Class Template Reference

```
#include <stream_iterator.h>
```

Inheritance diagram for std::ostream_iterator< _Tp, _CharT, _Traits >:



Public Types

- typedef [output_iterator_tag](#) `iterator_category`
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)
- using [difference_type](#) = `ptrdiff_t`
- typedef `_CharT` [char_type](#)
- typedef `_Traits` [traits_type](#)
- typedef [basic_ostream](#)< `_CharT`, `_Traits` > [ostream_type](#)

Public Member Functions

- [ostream_iterator](#) (const [ostream_iterator](#) &__obj) noexcept
- [ostream_iterator](#) ([ostream_type](#) &__s) noexcept
- [ostream_iterator](#) ([ostream_type](#) &__s, const `_CharT` *__c) noexcept
- [ostream_iterator](#) & **operator*** () noexcept
- [ostream_iterator](#) & **operator**++ () noexcept
- [ostream_iterator](#) & **operator**++ (int) noexcept
- [ostream_iterator](#) & **operator**= (const `_Tp` &__value)
- [ostream_iterator](#) & **operator**= (const [ostream_iterator](#) &)=default

6.777.1 Detailed Description

```
template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>>
class std::ostream_iterator<_Tp, _CharT, _Traits>
```

Provides output iterator semantics for streams.

This class provides an iterator to write to an ostream. The type Tp is the only type written by this iterator and there must be an operator<<(Tp) defined.

Template Parameters

<code>_Tp</code>	The type to write to the ostream.
<code>_CharT</code>	The ostream char_type.
<code>_Traits</code>	The ostream char_traits.

6.777.2 Member Typedef Documentation

char_type

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
typedef _CharT std::ostream_iterator<_Tp, _CharT, _Traits>::char_type
Public typedef.
```

difference_type

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
using std::ostream_iterator<_Tp, _CharT, _Traits>::difference_type = ptrdiff_t
Public typedef.
```

iterator_category

```
typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >↵
::iterator_category [inherited]
One of the tag types.
```

ostream_type

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
typedef basic_ostream<_CharT, _Traits> std::ostream_iterator<_Tp, _CharT, _Traits>::ostream_↵
type
Public typedef.
```

pointer

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer [inherited]
This type represents a pointer-to-value_type.
```

reference

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference [inherited]
This type represents a reference-to-value_type.
```

traits_type

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
typedef _Traits std::ostream_iterator< _Tp, _CharT, _Traits >::traits_type
Public typedef.
```

value_type

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type [inherited]
The type "pointed to" by the iterator.
```

6.777.3 Constructor & Destructor Documentation**ostream_iterator() [1/3]**

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator (
    ostream_type & __s ) [inline], [noexcept]
```

Construct from an ostream.

ostream_iterator() [2/3]

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator (
    ostream_type & __s,
    const _CharT * __c ) [inline], [noexcept]
```

Construct from an ostream.

The delimiter string *c* is written to the stream after every *Tp* written to the stream. The delimiter is not copied, and thus must not be destroyed while this iterator is in use.

Parameters

\leftarrow __s	Underlying ostream to write to.
\leftarrow __c	CharT delimiter string to insert.

ostream_iterator() [3/3]

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator (
    const ostream_iterator< _Tp, _CharT, _Traits > & __obj ) [inline], [noexcept]
```

Copy constructor.

6.777.4 Member Function Documentation**operator=()**

```
template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
ostream_iterator & std::ostream_iterator< _Tp, _CharT, _Traits >::operator= (
    const _Tp & __value ) [inline]
```

Writes *value* to underlying ostream using operator<<. If constructed with delimiter string, writes delimiter to ostream. The documentation for this class was generated from the following file:

- [stream_iterator.h](#)

6.778 `std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits>` Class Template Reference

Public Types

- typedef `_CharT` `char_type`
- typedef void `difference_type`
- typedef `output_iterator_tag` `iterator_category`
- typedef `basic_ostream<_CharT, _Traits>` `ostream_type`
- typedef void `pointer`
- typedef void `reference`
- typedef `_Traits` `traits_type`
- typedef void `value_type`

Public Member Functions

- `ostream_joiner` (`ostream_type` &__os, `_DelimT` &&__delimiter) noexcept(is_nothrow_move_constructible_v<_DelimT>)
- `ostream_joiner` (`ostream_type` &__os, const `_DelimT` &__delimiter) noexcept(is_nothrow_copy_constructible_v<_DelimT>)
- `ostream_joiner` & `operator*` () noexcept
- `ostream_joiner` & `operator++` () noexcept
- `ostream_joiner` & `operator++` (int) noexcept
- template<typename `_Tp`>
`ostream_joiner` & `operator=` (const `_Tp` &__value)

6.778.1 Detailed Description

```
template<typename _DelimT, typename _CharT = char, typename _Traits = char_traits<_CharT>>  
class std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits>
```

Output iterator that inserts a delimiter between elements.

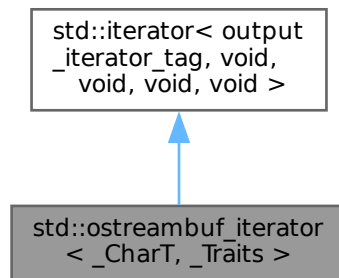
The documentation for this class was generated from the following file:

- [experimental/iterator](#)

6.779 `std::ostreambuf_iterator<_CharT, _Traits>` Class Template Reference

```
#include <ostreambuf_iterator.h>
```


Inheritance diagram for `std::ostreambuf_iterator< _CharT, _Traits >`:



Public Types

- typedef [output_iterator_tag](#) `iterator_category`
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)
- using [difference_type](#) = `ptrdiff_t`
- typedef `_CharT` [char_type](#)
- typedef `_Traits` [traits_type](#)
- typedef [basic_streambuf< _CharT, _Traits >](#) [streambuf_type](#)
- typedef [basic_ostream< _CharT, _Traits >](#) [ostream_type](#)

Public Member Functions

- [ostreambuf_iterator](#) ([ostream_type](#) &__s) noexcept
- [ostreambuf_iterator](#) ([streambuf_type](#) *__s) noexcept
- [ostreambuf_iterator](#) & [M_put](#) (const `_CharT` *__ws, [streamsize](#) __len)
- bool [failed](#) () const noexcept
- [ostreambuf_iterator](#) & [operator*](#) ()
- [ostreambuf_iterator](#) & [operator++](#) ()
- [ostreambuf_iterator](#) & [operator++](#) (int)
- [ostreambuf_iterator](#) & [operator=](#) (`_CharT` __c)

Friends

- template<typename `_CharT2` >
`__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, ostreambuf_iterator< _CharT2 >::__type` **copy**
`(istreambuf_iterator< _CharT2 >, istreambuf_iterator< _CharT2 >, ostreambuf_iterator< _CharT2 >)`

6.779.1 Detailed Description

```

template<typename _CharT, typename _Traits>
class std::ostreambuf_iterator< _CharT, _Traits >

```

Provides output iterator semantics for streambufs.

6.779.2 Member Typedef Documentation

char_type

```
template<typename _CharT , typename _Traits >
typedef _CharT std::ostreambuf_iterator< _CharT, _Traits >::char_type
Public typedefs.
```

difference_type

```
template<typename _CharT , typename _Traits >
using std::ostreambuf_iterator< _CharT, _Traits >::difference_type = ptrdiff_t
Public typedefs.
```

iterator_category

```
typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >←
::iterator_category [inherited]
One of the tag types.
```

ostream_type

```
template<typename _CharT , typename _Traits >
typedef basic_ostream<_CharT, _Traits> std::ostreambuf_iterator< _CharT, _Traits >::ostream_type
Public typedefs.
```

pointer

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer [inherited]
This type represents a pointer-to-value_type.
```

reference

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference [inherited]
This type represents a reference-to-value_type.
```

streambuf_type

```
template<typename _CharT , typename _Traits >
typedef basic_streambuf<_CharT, _Traits> std::ostreambuf_iterator< _CharT, _Traits >::streambuf←
_type
Public typedefs.
```

traits_type

```
template<typename _CharT , typename _Traits >
typedef _Traits std::ostreambuf_iterator< _CharT, _Traits >::traits_type
Public typedefs.
```

value_type

```
typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type [inherited]
The type "pointed to" by the iterator.
```

6.779.3 Constructor & Destructor Documentation

ostreambuf_iterator() [1/2]

```
template<typename _CharT , typename _Traits >
std::ostreambuf_iterator< _CharT, _Traits >::ostreambuf_iterator (
    ostream_type & __s ) [inline], [noexcept]
```

Construct output iterator from ostream.

ostreambuf_iterator() [2/2]

```
template<typename _CharT , typename _Traits >
std::ostreambuf_iterator< _CharT, _Traits >::ostreambuf_iterator (
    streambuf_type * __s ) [inline], [noexcept]
```

Construct output iterator from streambuf.

6.779.4 Member Function Documentation

failed()

```
template<typename _CharT , typename _Traits >
bool std::ostreambuf_iterator< _CharT, _Traits >::failed ( ) const [inline], [noexcept]
```

Return true if previous operator=() failed.

operator*()

```
template<typename _CharT , typename _Traits >
ostreambuf_iterator & std::ostreambuf_iterator< _CharT, _Traits >::operator* ( ) [inline]
```

Return *this.

operator++() [1/2]

```
template<typename _CharT , typename _Traits >
ostreambuf_iterator & std::ostreambuf_iterator< _CharT, _Traits >::operator++ ( ) [inline]
```

Return *this.

operator++() [2/2]

```
template<typename _CharT , typename _Traits >
ostreambuf_iterator & std::ostreambuf_iterator< _CharT, _Traits >::operator++ (
    int ) [inline]
```

Return *this.

operator=()

```
template<typename _CharT , typename _Traits >
ostreambuf_iterator & std::ostreambuf_iterator< _CharT, _Traits >::operator= (
    _CharT __c ) [inline]
```

Write character to streambuf. Calls streambuf.sputc().

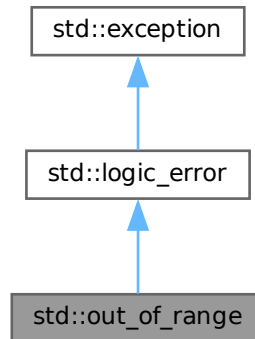
References [std::basic_streambuf< _CharT, _Traits >::sputc\(\)](#).

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [streambuf_iterator.h](#)

6.780 `std::out_of_range` Class Reference

Inheritance diagram for `std::out_of_range`:



Public Member Functions

- `out_of_range` (const char *) `_GLIBCXX_TXN_SAFE`
- `out_of_range` (const [out_of_range](#) &)=default
- `out_of_range` (const [string](#) &__arg) `_GLIBCXX_TXN_SAFE`
- `out_of_range` ([out_of_range](#) &&)=default
- `out_of_range` & `operator=` (const [out_of_range](#) &)=default
- `out_of_range` & `operator=` ([out_of_range](#) &&)=default
- virtual const char * `what` () const noexcept

6.780.1 Detailed Description

This represents an argument whose value is not within the expected range (e.g., boundary checks in `basic_string`).

6.780.2 Member Function Documentation

`what()`

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

6.781 `std::output_iterator_tag` Struct Reference

```
#include <stl_iterator_base_types.h>
```

6.781.1 Detailed Description

Marking output iterators.

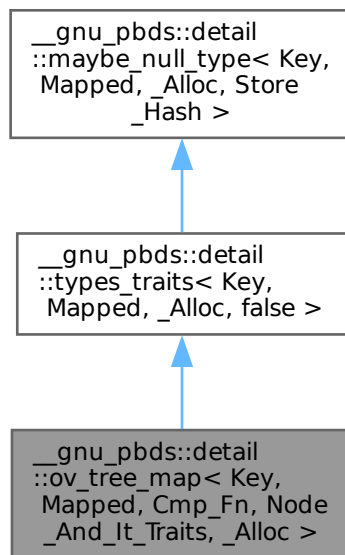
The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.782 `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

```
#include <ov_tree_map_.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`:



Classes

- class [cond_dtor](#)

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `point_const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `ov_tree_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `point_iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**

- `typedef traits_base::key_const_reference` **key_const_reference**
- `typedef traits_base::key_pointer` **key_pointer**
- `typedef traits_base::key_reference` **key_reference**
- `typedef traits_base::key_type` **key_type**
- `typedef traits_base::mapped_const_pointer` **mapped_const_pointer**
- `typedef traits_base::mapped_const_reference` **mapped_const_reference**
- `typedef traits_base::mapped_pointer` **mapped_pointer**
- `typedef traits_base::mapped_reference` **mapped_reference**
- `typedef traits_base::mapped_type` **mapped_type**
- `typedef __nothrowcopy::indicator` **no_throw_indicator**
- `typedef traits_type::node_const_iterator` **node_const_iterator**
- `typedef traits_type::node_iterator` **node_iterator**
- `typedef traits_type::node_update` **node_update**
- `typedef const_pointer` **point_const_iterator**
- `typedef pointer` **point_iterator**
- `typedef traits_base::pointer` **pointer**
- `typedef traits_base::reference` **reference**
- `typedef _Alloc::size_type` **size_type**
- `typedef integral_constant< int, Store_Hash >` **store_extra**
- `typedef stored_data< value_type, size_type, Store_Hash >` **stored_data_type**
- `typedef traits_base::value_type` **value_type**

Public Member Functions

- **ov_tree_map** (const Cmp_Fn &)
- **ov_tree_map** (const Cmp_Fn &, const node_update &)
- **ov_tree_map** (const [tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- iterator **erase** (iterator it)
- bool **erase** (key_const_reference)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference r_key)
- point_const_iterator **find** (key_const_reference r_key) const
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- [std::pair](#)< point_iterator, bool > **insert** (const_reference r_value)
- void **join** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **lower_bound** (key_const_reference r_key)
- point_const_iterator **lower_bound** (key_const_reference r_key) const
- size_type **max_size** () const
- node_iterator [node_begin](#) ()
- node_const_iterator [node_begin](#) () const
- node_iterator [node_end](#) ()

- node_const_iterator [node_end](#) () const
- mapped_reference **operator[]** (key_const_reference r_key)
- size_type **size** () const
- void **split** (key_const_reference, [tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- void **swap** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference r_key)
- point_const_iterator **upper_bound** (key_const_reference r_key) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

6.782.1 Detailed Description

template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _↵ Alloc>

class __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >

Ordered-vector tree associative-container.

6.782.2 Member Function Documentation

node_begin() [1/2]

```
template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >
```

```
node_iterator __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc
>::node_begin ( ) [inline]
```

Returns a node_iterator corresponding to the node at the root of the tree.

node_begin() [2/2]

```
template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >
```

```
node_const_iterator __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _↵
Alloc >::node_begin ( ) const [inline]
```

Returns a const node_iterator corresponding to the node at the root of the tree.

node_end() [1/2]

```
template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >
```

```
node_iterator __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc
>::node_end ( ) [inline]
```

Returns a node_iterator corresponding to a node just after a leaf of the tree.

node_end() [2/2]

```
template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >
```

```
node_const_iterator __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _↵
Alloc >::node_end ( ) const [inline]
```

Returns a const node_iterator corresponding to a node just after a leaf of the tree.

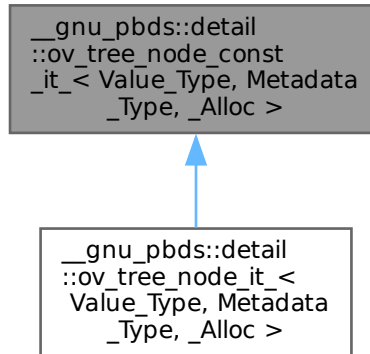
The documentation for this class was generated from the following file:

- [ov_tree_map_.hpp](#)

6.783 `__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >` Class Template Reference

`#include <node_iterators.hpp>`

Inheritance diagram for `__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >`:



Public Types

- typedef `rebind_traits< _Alloc, typename remove_const< Value_Type >::type >::const_pointer` **const_reference**
- typedef `trivial_iterator_difference_type` **difference_type**
- typedef `trivial_iterator_tag` **iterator_category**
- typedef `rebind_traits< _Alloc, metadata_type >::const_reference` **metadata_const_reference**
- typedef `Metadata_Type` **metadata_type**
- typedef `rebind_traits< _Alloc, typename remove_const< Value_Type >::type >::const_pointer` **reference**
- typedef `rebind_traits< _Alloc, Value_Type >::const_pointer` **value_type**

Public Member Functions

- `ov_tree_node_const_it_` (const_pointer p_nd=0, const_pointer p_begin_nd=0, const_pointer p_end_nd=0, const_metadata_pointer p_metadata=0)
- `this_type get_l_child` () const
- `metadata_const_reference get_metadata` () const
- `this_type get_r_child` () const
- bool **operator!=** (const `this_type` &other) const
- const_reference **operator*** () const
- bool **operator==** (const `this_type` &other) const

Public Attributes

- pointer `m_p_begin_value`
- pointer `m_p_end_value`
- const_metadata_pointer `m_p_metadata`
- pointer `m_p_value`

Protected Types

- typedef [rebind_traits](#)< _Alloc, Metadata_Type >::const_pointer **const_metadata_pointer**
- typedef [rebind_traits](#)< _Alloc, Value_Type >::const_pointer **const_pointer**
- typedef [rebind_traits](#)< _Alloc, Value_Type >::pointer **pointer**
- typedef [ov_tree_node_const_it](#)< Value_Type, Metadata_Type, _Alloc > **this_type**

Static Protected Member Functions

- template<typename Ptr >
static Ptr **mid_pointer** (Ptr p_begin, Ptr p_end)

6.783.1 Detailed Description

```
template<typename Value_Type, typename Metadata_Type, typename _Alloc>  
class __gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >
```

Const node reference.

6.783.2 Member Function Documentation

get_l_child()

```
template<typename Value_Type , typename Metadata_Type , typename _Alloc >  
this\_type __gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >::get_  
l_child ( ) const [inline]
```

Returns the node iterator associated with the left node.

get_r_child()

```
template<typename Value_Type , typename Metadata_Type , typename _Alloc >  
this\_type __gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >::get_  
r_child ( ) const [inline]
```

Returns the node iterator associated with the right node.

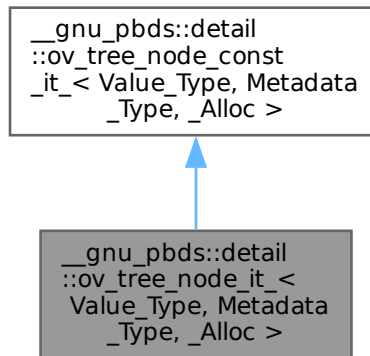
The documentation for this class was generated from the following file:

- [ov_tree_map_/node_iterators.hpp](#)

6.784 __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc > Class Template Reference

```
#include <node_iterators.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >`:



Public Types

- typedef [rebind_traits](#)< _Alloc, typename remove_const< Value_Type >::type >::pointer **const_reference**
- typedef [trivial_iterator_difference_type](#) **difference_type**
- typedef [trivial_iterator_tag](#) **iterator_category**
- typedef [rebind_traits](#)< _Alloc, metadata_type >::const_reference **metadata_const_reference**
- typedef Metadata_Type **metadata_type**
- typedef [rebind_traits](#)< _Alloc, typename remove_const< Value_Type >::type >::pointer **reference**
- typedef [rebind_traits](#)< _Alloc, Value_Type >::pointer **value_type**

Public Member Functions

- **ov_tree_node_it_** (const_pointer p_nd=0, const_pointer p_begin_nd=0, const_pointer p_end_nd=0, const_↵ metadata_pointer p_metadata=0)
- [ov_tree_node_it_get_l_child](#) () const
- metadata_const_reference **get_metadata** () const
- [ov_tree_node_it_get_r_child](#) () const
- bool **operator!=** (const [this_type](#) &other) const
- reference **operator*** () const
- bool **operator==** (const [this_type](#) &other) const

Public Attributes

- pointer **m_p_begin_value**
- pointer **m_p_end_value**
- const_metadata_pointer **m_p_metadata**
- pointer **m_p_value**

Static Protected Member Functions

- template<typename Ptr >
static Ptr **mid_pointer** (Ptr p_begin, Ptr p_end)

6.784.1 Detailed Description

```
template<typename Value_Type, typename Metadata_Type, typename _Alloc>
class __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc >
```

Node reference.

6.784.2 Member Function Documentation

get_l_child()

```
template<typename Value_Type , typename Metadata_Type , typename _Alloc >
ov_tree_node_it_ __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::get_l_
_l_child ( ) const [inline]
```

Returns the node reference associated with the left node.

get_r_child()

```
template<typename Value_Type , typename Metadata_Type , typename _Alloc >
ov_tree_node_it_ __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::get_r_
_r_child ( ) const [inline]
```

Returns the node reference associated with the right node.

operator*()

```
template<typename Value_Type , typename Metadata_Type , typename _Alloc >
reference __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::operator* (
) const [inline]
```

Access.

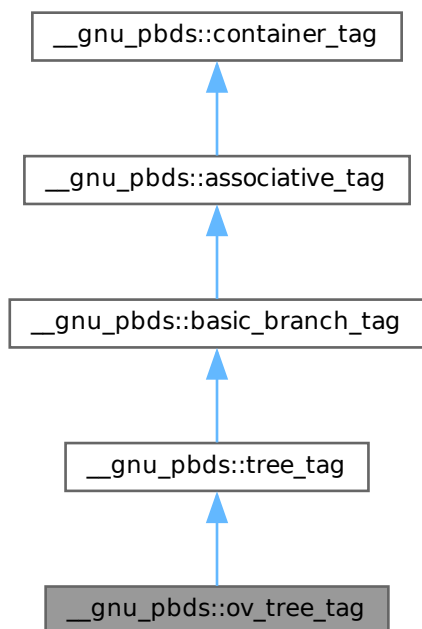
The documentation for this class was generated from the following file:

- [ov_tree_map_/node_iterators.hpp](#)

6.785 __gnu_pbds::ov_tree_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::ov_tree_tag:



6.785.1 Detailed Description

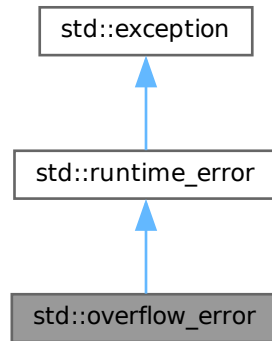
Ordered-vector tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.786 std::overflow_error Class Reference

Inheritance diagram for std::overflow_error:



Public Member Functions

- **overflow_error** (const char *) _GLIBCXX_TXN_SAFE
- **overflow_error** (const [overflow_error](#) &)=default
- **overflow_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **overflow_error** ([overflow_error](#) &&)=default
- [overflow_error](#) & **operator=** (const [overflow_error](#) &)=default
- [overflow_error](#) & **operator=** ([overflow_error](#) &&)=default
- virtual const char * [what](#) () const noexcept

6.786.1 Detailed Description

Thrown to indicate arithmetic overflow.

6.786.2 Member Function Documentation

what()

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

6.787 std::owner_less< _Tp > Struct Template Reference

6.787.1 Detailed Description

```
template<typename _Tp = void>
struct std::owner_less< _Tp >
```

Primary template owner_less.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

6.788 std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > > Struct Template Reference

```
#include <shared_ptr.h>
Inherits std::Sp_owner_less< _Tp, _Tp1 >.
```

Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- bool **operator()** (const _Tp &__lhs, const _Tp &__rhs) const noexcept
- bool **operator()** (const _Tp &__lhs, const _Tp1 &__rhs) const noexcept
- bool **operator()** (const _Tp1 &__lhs, const _Tp &__rhs) const noexcept

6.788.1 Detailed Description

```
template<typename _Tp>
struct std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >
```

Partial specialization of owner_less for shared_ptr.

6.788.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
typedef bool std::binary\_function< _Tp , _Tp , bool >::result_type [inherited]
result_type is the return type
```

second_argument_type

```
typedef _Tp std::binary\_function< _Tp , _Tp , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

6.789 `std::owner_less< shared_ptr< _Tp > >` Struct Template Reference

```
#include <shared_ptr.h>
Inherits std::_Sp_owner_less< _Tp, _Tp1 >.
```

Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `bool operator()` (`const _Tp &__lhs, const _Tp &__rhs`) `const noexcept`
- `bool operator()` (`const _Tp &__lhs, const _Tp1 &__rhs`) `const noexcept`
- `bool operator()` (`const _Tp1 &__lhs, const _Tp &__rhs`) `const noexcept`

6.789.1 Detailed Description

```
template<typename _Tp>
struct std::owner_less< shared_ptr< _Tp > >
```

Partial specialization of `owner_less` for `shared_ptr`.

6.789.2 Member Typedef Documentation

`first_argument_type`

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

`result_type`

```
typedef bool std::binary_function< _Tp , _Tp , bool >::result_type [inherited]
result_type is the return type
```

`second_argument_type`

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

6.790 `std::owner_less< void >` Struct Reference

```
#include <shared_ptr.h>
Inherits std::_Sp_owner_less< _Tp, _Tp1 >.
```

Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- bool **operator()** (const _Tp &__lhs, const _Tp &__rhs) const noexcept
- bool **operator()** (const _Tp &__lhs, const _Tp1 &__rhs) const noexcept
- bool **operator()** (const _Tp1 &__lhs, const _Tp &__rhs) const noexcept

6.790.1 Detailed Description

Void specialization of owner_less compares either shared_ptr or weak_ptr.

6.790.2 Member Typedef Documentation

first_argument_type

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::first_argument_type [inherited]
```

first_argument_type is the type of the first argument

result_type

```
typedef bool std::binary_function< _Tp , _Tp , bool >::result_type [inherited]
```

result_type is the return type

second_argument_type

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::second_argument_type [inherited]
```

second_argument_type is the type of the second argument

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

6.791 std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > > Struct Template Reference

```
#include <shared_ptr.h>
Inherits std::Sp_owner_less< _Tp, _Tp1 >.
```

Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- bool **operator()** (const _Tp &__lhs, const _Tp &__rhs) const noexcept
- bool **operator()** (const _Tp &__lhs, const _Tp1 &__rhs) const noexcept
- bool **operator()** (const _Tp1 &__lhs, const _Tp &__rhs) const noexcept

6.791.1 Detailed Description

template<typename _Tp>

struct std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >

Partial specialization of owner_less for weak_ptr.

6.791.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `bool` >::`first_argument_type` [inherited]
`first_argument_type` is the type of the first argument

result_type

typedef `bool` `std::binary_function`< `_Tp` , `_Tp` , `bool` >::`result_type` [inherited]
`result_type` is the return type

second_argument_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `bool` >::`second_argument_type` [inherited]
`second_argument_type` is the type of the second argument
The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

6.792 std::owner_less< weak_ptr< _Tp > > Struct Template Reference

#include <shared_ptr.h>
Inherits `std::Sp_owner_less`< `_Tp`, `_Tp1` >.

Public Types

- typedef `_Tp` `first_argument_type`
- typedef `bool` `result_type`
- typedef `_Tp` `second_argument_type`

Public Member Functions

- `bool` **operator()** (`const` `_Tp` &`__lhs`, `const` `_Tp` &`__rhs`) `const` `noexcept`
- `bool` **operator()** (`const` `_Tp` &`__lhs`, `const` `_Tp1` &`__rhs`) `const` `noexcept`
- `bool` **operator()** (`const` `_Tp1` &`__lhs`, `const` `_Tp` &`__rhs`) `const` `noexcept`

6.792.1 Detailed Description

template<typename `_Tp`>
struct `std::owner_less`< `weak_ptr`< `_Tp` > >

Partial specialization of `owner_less` for `weak_ptr`.

6.792.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `bool` >::`first_argument_type` [inherited]
`first_argument_type` is the type of the first argument

result_type

typedef `bool` `std::binary_function`< `_Tp` , `_Tp` , `bool` >::`result_type` [inherited]
`result_type` is the return type

second_argument_type

```
typedef _Tp std::binary_function< _Tp , _Tp , bool >::second_argument_type [inherited]
```

second_argument_type is the type of the second argument

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

6.793 std::packaged_task< _Res(_ArgTypes...)> Class Template Reference**Public Member Functions**

- `template<typename _Fn, typename = __not_same<_Fn>>`
packaged_task (`_Fn &&__fn`)
- **packaged_task** (`const packaged_task &`)=delete
- **packaged_task** (`packaged_task &&__other`) noexcept
- `future<_Res>` **get_future** ()
- void **make_ready_at_thread_exit** (`_ArgTypes... __args`)
- void **operator()** (`_ArgTypes... __args`)
- `packaged_task &` **operator=** (`const packaged_task &`)=delete
- `packaged_task &` **operator=** (`packaged_task &&__other`) noexcept
- void **reset** ()
- void **swap** (`packaged_task &__other`) noexcept
- bool **valid** () const noexcept

6.793.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>
```

```
class std::packaged_task< _Res(_ArgTypes...)>
```

packaged_task

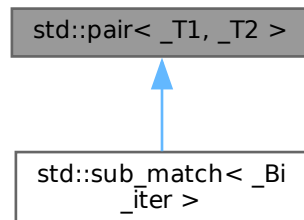
The documentation for this class was generated from the following file:

- [future](#)

6.794 std::pair< _T1, _T2 > Struct Template Reference

```
#include <stl_pair.h>
```

Inheritance diagram for std::pair< _T1, _T2 >:



Public Types

- typedef `_T1` [first_type](#)
- typedef `_T2` [second_type](#)

Public Member Functions

- template<typename `_U1` = `_T1`, typename `_U2` = `_T2`, typename [enable_if](#)< `__and`< `__is_implicitly_default_constructible`< `_U1` >, `__is_implicitly_default_constructible`< `_U2` > >::value, bool >::type = true>
constexpr [pair](#) ()
- template<typename `_U2`, `__enable_if_t`< `__and`< [is_pointer](#)< `_T1` >, `__not`< [is_reference](#)< `_U2` > >, [is_constructible](#)< `_T2`, `_U2` >, `__not`< [is_constructible](#)< `_T2`, const `_U2` & > >, [is_convertible](#)< `_U2`, `_T2` > >::value, bool > = true>
constexpr [pair](#) (`_zero_as_null_pointer_constant`, `_U2` && `_y`,...)
- template<typename `_U2`, `__enable_if_t`< `__and`< [is_pointer](#)< `_T1` >, `__not`< [is_reference](#)< `_U2` > >, [is_constructible](#)< `_T2`, `_U2` >, `__not`< [is_constructible](#)< `_T2`, const `_U2` & > >, `__not`< [is_convertible](#)< `_U2`, `_T2` > > >::value, bool > = false>
constexpr [pair](#) (`_zero_as_null_pointer_constant`, `_U2` && `_y`,...)
- template<typename `_U1`, `__enable_if_t`< `__and`< `__not`< [is_reference](#)< `_U1` > >, [is_pointer](#)< `_T2` >, [is_constructible](#)< `_T1`, `_U1` >, `__not`< [is_constructible](#)< `_T1`, const `_U1` & > >, [is_convertible](#)< `_U1`, `_T1` > > >::value, bool > = true>
constexpr [pair](#) (`_U1` && `_x`, `_zero_as_null_pointer_constant`,...)
- template<typename `_U1`, `__enable_if_t`< `__and`< `__not`< [is_reference](#)< `_U1` > >, [is_pointer](#)< `_T2` >, [is_constructible](#)< `_T1`, `_U1` >, `__not`< [is_constructible](#)< `_T1`, const `_U1` & > >, `__not`< [is_convertible](#)< `_U1`, `_T1` > > >::value, bool > = false>
constexpr [pair](#) (`_U1` && `_x`, `_zero_as_null_pointer_constant`,...)
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCP`::template `_MoveConstructiblePair`< `_U1`, `_U2` >() && `_PCCP`↵
::template `_ImplicitlyMoveConvertiblePair`< `_U1`, `_U2` >(), bool >::type = true>
constexpr [pair](#) (`_U1` && `_x`, `_U2` && `_y`)
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCP`::template `_MoveConstructiblePair`< `_U1`, `_U2` >() && !`_PCCP`↵
::template `_ImplicitlyMoveConvertiblePair`< `_U1`, `_U2` >(), bool >::type = false>
constexpr [pair](#) (`_U1` && `_x`, `_U2` && `_y`)
- template<typename `_U1` = `_T1`, typename `_U2` = `_T2`, typename [enable_if](#)< `_PCCP`::template `_ConstructiblePair`< `_U1`, `_U2` >() && ↵
`_PCCP`::template `_ImplicitlyConvertiblePair`< `_U1`, `_U2` >(), bool >::type = true>
constexpr [pair](#) (const `_T1` & `_a`, const `_T2` & `_b`)
- template<typename `_U1` = `_T1`, typename `_U2` = `_T2`, typename [enable_if](#)< `_PCCP`::template `_ConstructiblePair`< `_U1`, `_U2` >() && !↵
`_PCCP`::template `_ImplicitlyConvertiblePair`< `_U1`, `_U2` >(), bool >::type = false>
constexpr [pair](#) (const `_T1` & `_a`, const `_T2` & `_b`)
- constexpr [pair](#) (const [pair](#) &)=default
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCFP`< `_U1`, `_U2` >::template `_ConstructiblePair`< `_U1`, `_U2` >() && `_PCCFP`< `_U1`, `_U2` >::template `_ImplicitlyConvertiblePair`< `_U1`, `_U2` >(), bool >::type = true>
constexpr [pair](#) (const [pair](#)< `_U1`, `_U2` > & `_p`)
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCFP`< `_U1`, `_U2` >::template `_ConstructiblePair`< `_U1`, `_U2` >() && !`_PCCFP`< `_U1`, `_U2` >::template `_ImplicitlyConvertiblePair`< `_U1`, `_U2` >(), bool >::type = false>
constexpr [pair](#) (const [pair](#)< `_U1`, `_U2` > & `_p`)
- constexpr [pair](#) ([pair](#) &&)=default
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCFP`< `_U1`, `_U2` >::template `_MoveConstructiblePair`< `_U1`, `_U2` >() && `_PCCFP`< `_U1`, `_U2` >::template `_ImplicitlyMoveConvertiblePair`< `_U1`, `_U2` >(), bool >::type = true>
constexpr [pair](#) ([pair](#)< `_U1`, `_U2` > && `_p`)
- template<typename `_U1`, typename `_U2`, typename [enable_if](#)< `_PCCFP`< `_U1`, `_U2` >::template `_MoveConstructiblePair`< `_U1`, `_U2` >() && !`_PCCFP`< `_U1`, `_U2` >::template `_ImplicitlyMoveConvertiblePair`< `_U1`, `_U2` >(), bool >::type = false>
constexpr [pair](#) ([pair](#)< `_U1`, `_U2` > && `_p`)
- template<typename... `_Args1`, typename... `_Args2`>
constexpr [pair](#) ([piecewise_construct_t](#), `tuple`< `_Args1`... >, `tuple`< `_Args2`... >)
- [pair](#) & **operator=** (`__conditional_t`< `__and`< [is_copy_assignable](#)< `_T1` >, [is_copy_assignable](#)< `_T2` > >↵
::value, const [pair](#) &, const `__nonexistent` & > `_p`)

- `pair` & `operator=` (`__conditional_t< __and_< is_move_assignable< _T1 >, is_move_assignable< _T2 > >::value, pair &&, __nonexistent && > __p`) `noexcept(__and_< is_nothrow_move_assignable< _T1 >, is_nothrow_move_assignable< _T2 > >::value)`
- `template<typename _U1, typename _U2 >`
`enable_if< __and_< is_assignable< _T1 &, const _U1 & >, is_assignable< _T2 &, const _U2 & > >::value, pair & >::type` `operator=` (`const pair< _U1, _U2 > & __p`)
- `template<typename _U1, typename _U2 >`
`enable_if< __and_< is_assignable< _T1 &, _U1 && >, is_assignable< _T2 &, _U2 && > >::value, pair & >::type` `operator=` (`pair< _U1, _U2 > && __p`)
- `constexpr void swap` (`pair & __p`) `noexcept(__and_< __is_nothrow_swappable< _T1 >, __is_nothrow_swappable< _T2 > >::value)`

Public Attributes

- `_T1` *first*
- `_T2` *second*

Related Symbols

(Note that these are not member symbols.)

- `template<typename _T1, typename _T2 >`
`constexpr pair< typename __decay_and_strip< _T1 >::__type, typename __decay_and_strip< _T2 >::__type >` `make_pair` (`_T1 && __x, _T2 && __y`)
- `template<typename _T1, typename _T2 >`
`pair` (`_T1, _T2`) -> `pair< _T1, _T2 >`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator==` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr bool operator<` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr bool operator!=` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr bool operator>` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr bool operator<=` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr bool operator>=` (`const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y`)
- `template<typename _T1, typename _T2 >`
`constexpr enable_if< __and_< __is_swappable< _T1 >, __is_swappable< _T2 > >::value >::type` `swap` (`pair< _T1, _T2 > & __x, pair< _T1, _T2 > & __y`) `noexcept(noexcept(__x.swap(__y)))`

6.794.1 Detailed Description

`template<typename _T1, typename _T2>`
`struct std::pair<_T1, _T2>`

Struct holding two objects of arbitrary type.

Template Parameters

<code>_T1</code>	Type of first object.
<code>_T2</code>	Type of second object.

<https://gcc.gnu.org/onlinedocs/libstdc++/manual/utilities.html>

6.794.2 Member Typedef Documentation

first_type

```
template<typename _T1 , typename _T2 >
typedef _T1 std::pair< _T1, _T2 >::first_type
```

The type of the first member.

second_type

```
template<typename _T1 , typename _T2 >
typedef _T2 std::pair< _T1, _T2 >::second_type
```

The type of the second member.

6.794.3 Constructor & Destructor Documentation

pair() [1/5]

```
template<typename _T1 , typename _T2 >
constexpr std::pair< _T1, _T2 >::pair (
    const pair< _T1, _T2 > & ) [constexpr], [default]
```

Copy constructor.

pair() [2/5]

```
template<typename _T1 , typename _T2 >
constexpr std::pair< _T1, _T2 >::pair (
    pair< _T1, _T2 > && ) [constexpr], [default]
```

Move constructor.

pair() [3/5]

```
template<typename _T1 , typename _T2 >
template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< __and< __is_implicitly_↔
default_constructible< _U1 >, __is_implicitly_default_constructible< _U2 > > ::value, bool >↔
::type = true>
constexpr std::pair< _T1, _T2 >::pair ( ) [inline], [constexpr]
```

The default constructor creates first and second using their respective default constructors.

pair() [4/5]

```
template<typename _T1 , typename _T2 >
template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< _PCCP::template _Constructible↔
Pair< _U1, _U2 >() &&_PCCP::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type =
true>
constexpr std::pair< _T1, _T2 >::pair (
    const _T1 & __a,
    const _T2 & __b ) [inline], [constexpr]
```

Construct from two const lvalues, allowing implicit conversions.

pair() [5/5]

```
template<typename _T1 , typename _T2 >
template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< _PCCP::template _ConstructiblePair<
Pair< _U1, _U2 >() &&!_PCCP::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type =
false>
constexpr std::pair< _T1, _T2 >::pair (
    const _T1 & __a,
    const _T2 & __b ) [inline], [explicit], [constexpr]
```

Construct from two const lvalues, disallowing implicit conversions.

6.794.4 Member Function Documentation**swap()**

```
template<typename _T1 , typename _T2 >
constexpr void std::pair< _T1, _T2 >::swap (
    pair< _T1, _T2 > & __p ) [inline], [constexpr], [noexcept]
```

Swap the first members and then the second members.

References `std::pair< _T1, _T2 >::first`, `std::pair< _T1, _T2 >::second`, `std::swap()`, and `std::pair< _T1, _T2 >::swap()`.

Referenced by `std::swap()`, `std::swap()`, `std::pair< _T1, _T2 >::swap()`, `std::pair< _T1, _T2 >::swap()`, and `std::sub_match< _Biter >::swap()`.

6.794.5 Member Data Documentation**first**

```
template<typename _T1 , typename _T2 >
_T1 std::pair< _T1, _T2 >::first
```

The first member.

Referenced by `std::_Temporary_buffer< _ForwardIterator, _Tp >::_Temporary_buffer()`, `__gnu_parallel::_find_template()`, `__gnu_parallel::_find_template()`, `std::_sample()`, `std::set< _Key, _Compare, _Alloc >::insert()`, `std::pair< _T1, _T2 >::operator<()`, `std::pair< _T1, _T2 >::operator==()`, `std::shuffle()`, and `std::pair< _T1, _T2 >::swap()`.

second

```
template<typename _T1 , typename _T2 >
_T2 std::pair< _T1, _T2 >::second
```

The second member.

Referenced by `std::_Temporary_buffer< _ForwardIterator, _Tp >::_Temporary_buffer()`, `std::_sample()`, `std::set< _Key, _Compare, _Alloc >::insert()`, `std::pair< _T1, _T2 >::operator<()`, `std::pair< _T1, _T2 >::operator==()`, `std::shuffle()`, and `std::pair< _T1, _T2 >::swap()`.

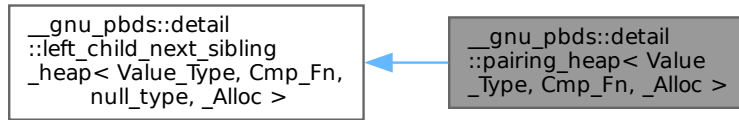
The documentation for this struct was generated from the following files:

- `bits/stl_iterator.h`
- `stl_pair.h`
- `tuple`

6.795 `__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

```
#include <pairing_heap.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_a::const_pointer` **const_pointer**
- typedef `__rebind_a::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `left_child_next_sibling_heap_node_< Value_Type, null_type, _Alloc >` **node**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_a::pointer` **pointer**
- typedef `__rebind_a::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **pairing_heap** (`const Cmp_Fn &`)
- **pairing_heap** (`const pairing_heap &`)
- **iterator begin** ()
- **const_iterator begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator end** ()
- **const_iterator end** () const
- void **erase** (`point_iterator`)
- template<typename Pred >
size_type **erase_if** (Pred)
- `Cmp_Fn &` **get_cmp_fn** ()
- `const Cmp_Fn &` **get_cmp_fn** () const
- void **join** (`pairing_heap &`)
- size_type **max_size** () const
- void **modify** (`point_iterator`, `const_reference`)
- void **pop** ()
- `point_iterator` **push** (`const_reference`)
- size_type **size** () const

- `template<typename Pred >`
`void split (Pred, pairing_heap &)`
- `void swap (left_child_next_sibling_heap< Value_Type, Cmp_Fn, null_type, _Alloc > &)`
- `void swap (pairing_heap &)`
- `const_reference top () const`

Protected Types

- `typedef alloc_traits::allocator_type node_allocator`
- `typedef alloc_traits::const_pointer node_const_pointer`
- `typedef null_type node_metadata`
- `typedef std::pair< node_pointer, node_pointer > node_pointer_pair`

Protected Member Functions

- `void actual_erase_node (node_pointer)`
- `void bubble_to_top (node_pointer)`
- `void clear_imp (node_pointer)`
- `template<typename It >`
`void copy_from_range (It, It)`
- `node_pointer get_new_node_for_insert (const_reference)`
- `node_pointer prune (Pred)`
- `void swap_with_parent (node_pointer, node_pointer)`
- `void to_linked_list ()`
- `void value_swap (left_child_next_sibling_heap &)`

Static Protected Member Functions

- `static void make_child_of (node_pointer, node_pointer)`
- `static node_pointer parent (node_pointer)`

Protected Attributes

- `node_pointer m_p_root`
- `size_type m_size`

6.795.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >
```

Pairing heap.

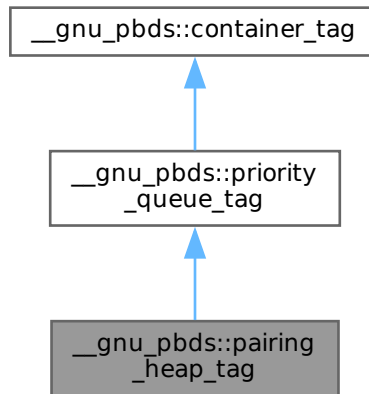
The documentation for this class was generated from the following file:

- [pairing_heap.hpp](#)

6.796 `__gnu_pbds::pairing_heap_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```


Inheritance diagram for `__gnu_pbds::pairing_heap_tag`:



6.796.1 Detailed Description

Pairing-heap.

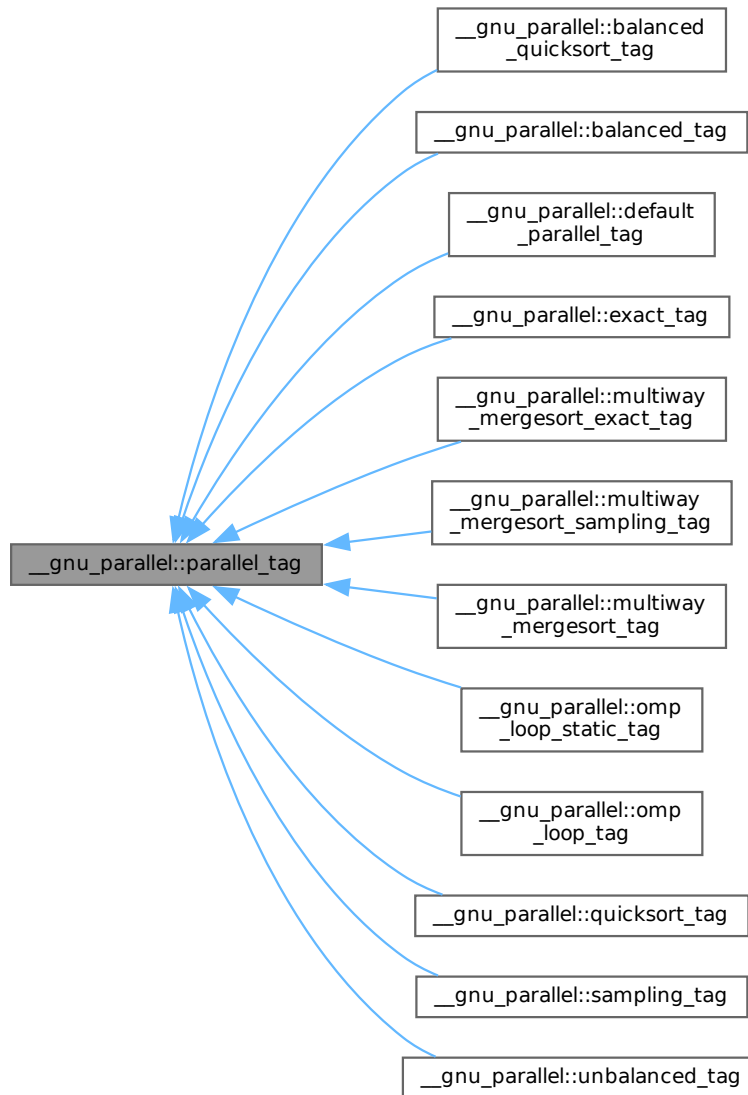
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.797 `__gnu_parallel::parallel_tag` Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::parallel_tag:



Public Member Functions

- [parallel_tag](#) ()
- [parallel_tag](#) ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) [__get_num_threads](#) ()
- void [set_num_threads](#) ([_ThreadIndex](#) __num_threads)

6.797.1 Detailed Description

Recommends parallel execution at compile time, optionally using a user-specified number of threads.

6.797.2 Constructor & Destructor Documentation

`parallel_tag()` [1/2]

```
__gnu_parallel::parallel_tag::parallel_tag ( ) [inline]
```

Default constructor. Use default number of threads.

`parallel_tag()` [2/2]

```
__gnu_parallel::parallel_tag::parallel_tag (
    __ThreadIndex __num_threads ) [inline]
```

Default constructor. Recommend number of threads to use.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

6.797.3 Member Function Documentation

`__get_num_threads()`

```
__ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

```
void __gnu_parallel::parallel_tag::set_num_threads (
    __ThreadIndex __num_threads ) [inline]
```

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.798 `std::bernoulli_distribution::param_type` Struct Reference

```
#include <random.h>
```

Public Types

- typedef [bernoulli_distribution](#) `distribution_type`

Public Member Functions

- `param_type` (double __p)
- double `p` () const

Friends

- `bool operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- `bool operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.798.1 Detailed Description

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.799 `std::binomial_distribution<_IntType>::param_type` Struct Reference

```
#include <random.h>
```

Public Types

- typedef [binomial_distribution<_IntType>](#) **distribution_type**

Public Member Functions

- **param_type** (`_IntType` __t, double __p=0.5)
- double **p** () const
- `_IntType` **t** () const

Friends

- class **binomial_distribution<_IntType>**
- `bool operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- `bool operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.799.1 Detailed Description

```
template<typename _IntType = int>
```

```
struct std::binomial_distribution<_IntType>::param_type
```

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.800 `std::cauchy_distribution<_RealType>::param_type` Struct Reference

```
#include <random.h>
```

Public Types

- typedef [cauchy_distribution<_RealType>](#) **distribution_type**

Public Member Functions

- **param_type** (`_RealType` __a, `_RealType` __b=_RealType(1))
- `_RealType` **a** () const
- `_RealType` **b** () const

Friends

- `bool operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- `bool operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.800.1 Detailed Description

```
template<typename _RealType = double>
struct std::cauchy_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.801 std::chi_squared_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [chi_squared_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- [param_type](#) (_RealType __n)
- [_RealType n](#) () const

Friends

- `bool operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- `bool operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.801.1 Detailed Description

```
template<typename _RealType = double>
struct std::chi_squared_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.802 std::discrete_distribution< _IntType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [discrete_distribution](#)< _IntType > **distribution_type**

Public Member Functions

- `template<typename _InputIterator >`
`param_type` (_InputIterator __wbegin, _InputIterator __wend)
- `param_type` (const [param_type](#) &)=default
- `param_type` ([initializer_list](#)< double > __wil)

- template<typename _Func >
 param_type (size_t __nw, double __xmin, double __xmax, _Func __fw)
- [param_type](#) & **operator=** (const [param_type](#) &)=default
- [std::vector](#)< double > **probabilities** () const

Friends

- class **discrete_distribution**< _IntType >
- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.802.1 Detailed Description

template<typename _IntType = int>
struct std::discrete_distribution< _IntType >::param_type

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.803 std::exponential_distribution<_RealType>::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [exponential_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __lambda)
- _RealType **lambda** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.803.1 Detailed Description

template<typename _RealType = double>
struct std::exponential_distribution< _RealType >::param_type

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.804 std::extreme_value_distribution<_RealType>::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [extreme_value_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __a, _RealType __b=_RealType(1.0))
- _RealType **a** () const
- _RealType **b** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.804.1 Detailed Description

```
template<typename _RealType = double>
struct std::extreme_value_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.805 std::fisher_f_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [fisher_f_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __m, _RealType __n=_RealType(1))
- _RealType **m** () const
- _RealType **n** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.805.1 Detailed Description

```
template<typename _RealType = double>
struct std::fisher_f_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.806 std::gamma_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [gamma_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __alpha_val, _RealType __beta_val=_RealType(1))
- _RealType **alpha** () const
- _RealType **beta** () const

Friends

- class **gamma_distribution**< _RealType >
- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.806.1 Detailed Description

```
template<typename _RealType = double>
struct std::gamma_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.807 std::geometric_distribution< _IntType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [geometric_distribution](#)< _IntType > **distribution_type**

Public Member Functions

- **param_type** (double __p)
- double **p** () const

Friends

- class **geometric_distribution**< _IntType >
- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.807.1 Detailed Description

```
template<typename _IntType = int>
struct std::geometric_distribution< _IntType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.808 std::lognormal_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```


Public Types

- typedef [lognormal_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __m, _RealType __s=_RealType(1))
- _RealType **m** () const
- _RealType **s** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.808.1 Detailed Description

```
template<typename _RealType = double>
struct std::lognormal_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.809 std::negative_binomial_distribution< _IntType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [negative_binomial_distribution](#)< _IntType > **distribution_type**

Public Member Functions

- **param_type** (_IntType __k, double __p=0.5)
- _IntType **k** () const
- double **p** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.809.1 Detailed Description

```
template<typename _IntType = int>
struct std::negative_binomial_distribution< _IntType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.810 std::normal_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [normal_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __mean, _RealType __stddev=_RealType(1))
- _RealType **mean** () const
- _RealType **stddev** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.810.1 Detailed Description

template<typename _RealType = double>
struct std::normal_distribution< _RealType >::param_type

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.811 std::piecewise_constant_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [piecewise_constant_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- template<typename _InputIteratorB, typename _InputIteratorW >
param_type (_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin)
- **param_type** (const [param_type](#) &)=default
- template<typename _Func >
param_type ([initializer_list](#)< _RealType > __bi, _Func __fw)
- template<typename _Func >
param_type (size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw)
- [std::vector](#)< double > **densities** () const
- [std::vector](#)< _RealType > **intervals** () const
- [param_type](#) & **operator=** (const [param_type](#) &)=default

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- class [piecewise_constant_distribution](#)< _RealType >

6.811.1 Detailed Description

```
template<typename _RealType = double>
struct std::piecewise_constant_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.812 std::piecewise_linear_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [piecewise_linear_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- template<typename _InputIteratorB, typename _InputIteratorW >
param_type (_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin)
- **param_type** (const [param_type](#) &)=default
- template<typename _Func >
param_type ([initializer_list](#)< _RealType > __bl, _Func __fw)
- template<typename _Func >
param_type (size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw)
- [std::vector](#)< double > **densities** () const
- [std::vector](#)< _RealType > **intervals** () const
- [param_type](#) & **operator=** (const [param_type](#) &)=default

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- class [piecewise_linear_distribution](#)< _RealType >

6.812.1 Detailed Description

```
template<typename _RealType = double>
struct std::piecewise_linear_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.813 std::poisson_distribution< _IntType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [poisson_distribution](#)< _IntType > **distribution_type**

Public Member Functions

- `param_type` (double __mean)
- double `mean` () const

Friends

- bool `operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool `operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- class `poisson_distribution<_IntType>`

6.813.1 Detailed Description

```
template<typename _IntType = int>
struct std::poisson_distribution<_IntType>::param_type
```

Parameter type.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.814 `std::student_t_distribution<_RealType>::param_type` Struct Reference

```
#include <random.h>
```

Public Types

- typedef [student_t_distribution<_RealType>](#) `distribution_type`

Public Member Functions

- `param_type` (_RealType __n)
- _RealType `n` () const

Friends

- bool `operator!=` (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool `operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.814.1 Detailed Description

```
template<typename _RealType = double>
struct std::student_t_distribution<_RealType>::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.815 `std::uniform_int_distribution<_IntType>::param_type` Struct Reference

```
#include <uniform_int_dist.h>
```

Public Types

- typedef [uniform_int_distribution<_IntType>](#) `distribution_type`

Public Member Functions

- **param_type** (`_IntType __a, _IntType __b= __gnu_cxx::__int_traits< _IntType >::__max`)
- **result_type a** () const
- **result_type b** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.815.1 Detailed Description

```
template<typename _IntType = int>
struct std::uniform_int_distribution< _IntType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [uniform_int_dist.h](#)

6.816 std::uniform_real_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [uniform_real_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (`_RealType __a, _RealType __b=_RealType(1)`)
- **result_type a** () const
- **result_type b** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.816.1 Detailed Description

```
template<typename _RealType = double>
struct std::uniform_real_distribution< _RealType >::param_type
```

Parameter type.

The documentation for this struct was generated from the following file:

- [random.h](#)

6.817 std::weibull_distribution< _RealType >::param_type Struct Reference

```
#include <random.h>
```

Public Types

- typedef [weibull_distribution](#)< _RealType > **distribution_type**

Public Member Functions

- **param_type** (_RealType __a, _RealType __b=_RealType(1.0))
- _RealType **a** () const
- _RealType **b** () const

Friends

- bool **operator!=** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

6.817.1 Detailed Description

template<typename _RealType = double>
 struct std::weibull_distribution< _RealType >::param_type

Parameter type.

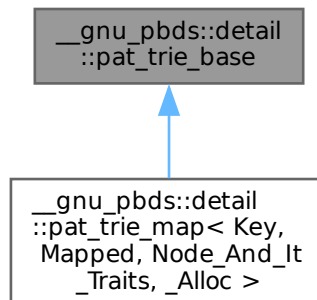
The documentation for this struct was generated from the following file:

- [random.h](#)

6.818 __gnu_pbds::detail::pat_trie_base Struct Reference

#include <pat_trie_base.hpp>

Inheritance diagram for __gnu_pbds::detail::pat_trie_base:

**Classes**

- class [_CIter](#)
- struct [_Head](#)
- struct [_Inode](#)
- class [_Iter](#)
- struct [_Leaf](#)
- struct [_Metadata](#)
- struct [_Metadata< null_type, _Alloc >](#)
- struct [_Node_base](#)
- class [_Node_citer](#)
- class [_Node_iter](#)

Public Types

- enum `node_type` { `i_node` , `leaf_node` , `head_node` }

6.818.1 Detailed Description

Base type for PATRICIA trees.

6.818.2 Member Enumeration Documentation

`node_type`

enum `__gnu_pbds::detail::pat_trie_base::node_type`

Three types of nodes.

`i_node` is used by `_Inode`, `leaf_node` by `_Leaf`, and `head_node` by `_Head`.

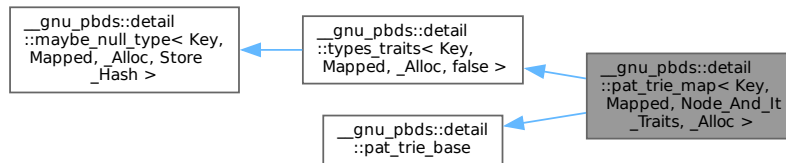
The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

6.819 `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >` Class Template Reference

```
#include <pat_trie.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >`:



Public Types

- typedef `traits_type::access_traits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `point_const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `traits_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `pat_trie_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `point_iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**
- typedef `traits_base::key_const_reference` **key_const_reference**
- typedef `traits_base::key_pointer` **key_pointer**
- typedef `traits_base::key_reference` **key_reference**
- typedef `traits_base::key_type` **key_type**
- typedef `traits_base::mapped_const_pointer` **mapped_const_pointer**

- typedef traits_base::mapped_const_reference **mapped_const_reference**
- typedef traits_base::mapped_pointer **mapped_pointer**
- typedef traits_base::mapped_reference **mapped_reference**
- typedef traits_base::mapped_type **mapped_type**
- typedef __nothrowcopy::indicator **no_throw_indicator**
- typedef traits_type::node_const_iterator **node_const_iterator**
- typedef traits_type::node_iterator **node_iterator**
- enum [node_type](#) { [i_node](#) , [leaf_node](#) , [head_node](#) }
- typedef traits_type::node_update **node_update**
- typedef traits_type::const_iterator **point_const_iterator**
- typedef traits_type::iterator **point_iterator**
- typedef traits_base::pointer **pointer**
- typedef traits_base::reference **reference**
- typedef traits_type::reverse_iterator **reverse_iterator**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef [stored_data](#)< [value_type](#), size_type, Store_Hash > **stored_data_type**
- typedef [traits_base::value_type](#) **value_type**

Public Member Functions

- **pat_trie_map** (const access_traits &)
- **pat_trie_map** (const [tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- const_iterator **erase** (const_iterator)
- const_reverse_iterator **erase** (const_reverse_iterator)
- iterator **erase** (iterator)
- bool **erase** (key_const_reference)
- reverse_iterator **erase** (reverse_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- access_traits & **get_access_traits** ()
- const access_traits & **get_access_traits** () const
- node_update & **get_node_update** ()
- const node_update & **get_node_update** () const
- [std::pair](#)< point_iterator, bool > **insert** (const_reference)
- void **join** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **lower_bound** (key_const_reference)
- point_const_iterator **lower_bound** (key_const_reference) const
- size_type **max_size** () const
- node_iterator [node_begin](#) ()
- node_const_iterator [node_begin](#) () const
- node_iterator [node_end](#) ()
- node_const_iterator [node_end](#) () const

- mapped_reference **operator[]** (key_const_reference r_key)
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- size_type **size** () const
- void **split** (key_const_reference, tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- void **swap** (tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference)
- point_const_iterator **upper_bound** (key_const_reference) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Member Functions

- template<typename It >
void **copy_from_range** (It, It)
- node_pointer **recursive_copy_node** (node_const_pointer)
- void **value_swap** (tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)

6.819.1 Detailed Description

template<typename Key, typename Mapped, typename Node_And_It_Traits, typename _Alloc>
class **__gnu_pbds::detail::pat_trie_map**< Key, Mapped, Node_And_It_Traits, _Alloc >

PATRICIA trie.

This implementation loosely borrows ideas from: 1) Fast Mergeable Integer Maps, Okasaki, Gill 1998 2) Pset: Sets of integers implemented as Patricia trees, Jean-Christophe Filliatr, 2000

6.819.2 Member Enumeration Documentation

node_type

enum **__gnu_pbds::detail::pat_trie_base::node_type** [inherited]

Three types of nodes.

i_node is used by _Inode, leaf_node by _Leaf, and head_node by _Head.

6.819.3 Member Function Documentation

node_begin() [1/2]

```
template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc >
node_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node↵
_begin ( ) [inline]
```

Returns a node_iterator corresponding to the node at the root of the tree.

node_begin() [2/2]

```
template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc >
node_const_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >↵
::node_begin ( ) const [inline]
```

Returns a const node_iterator corresponding to the node at the root of the tree.

node_end() [1/2]

```
template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc >
node_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node↔
_end ( ) [inline]
```

Returns a node_iterator corresponding to a node just after a leaf of the tree.

node_end() [2/2]

```
template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc >
node_const_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >↔
::node_end ( ) const [inline]
```

Returns a const node_iterator corresponding to a node just after a leaf of the tree.

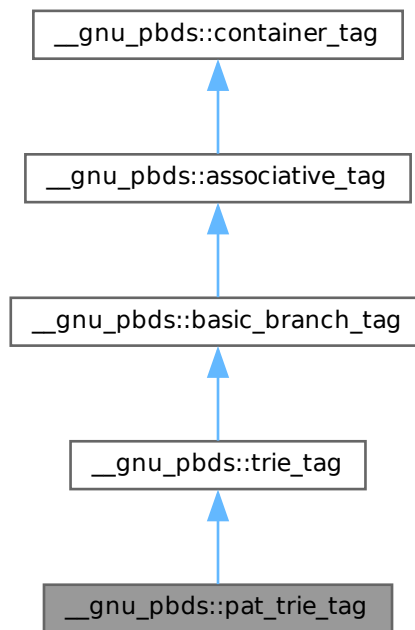
The documentation for this class was generated from the following file:

- [pat_trie_.hpp](#)

6.820 __gnu_pbds::pat_trie_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::pat_trie_tag:

**6.820.1 Detailed Description**

PATRICIA trie.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.821 std::experimental::filesystem::v1::path Class Reference

```
#include <fs_path.h>
```

Classes

- class [iterator](#)

Public Types

- typedef [iterator](#) **const_iterator**
- typedef [std::basic_string](#)< value_type > **string_type**
- typedef char **value_type**

Public Member Functions

- template<typename _InputIterator, typename _Require = __detail::_Path<_InputIterator, _InputIterator>>>
path (_InputIterator __first, _InputIterator __last)
- template<typename _InputIterator, typename _Require = __detail::_Path<_InputIterator, _InputIterator>, typename _Require2 = __detail::__value_type_is_char<_InputIterator>>>
path (_InputIterator __first, _InputIterator __last, const [locale](#) &__loc)
- template<typename _Source, typename _Require = __detail::_Path<_Source>>>
path (_Source const &__source)
- template<typename _Source, typename _Require = __detail::_Path<_Source>, typename _Require2 = __detail::__value_type_is_char<_Source>>>
path (_Source const &__source, const [locale](#) &__loc)
- **path** (const [path](#) &__p)
- **path** ([path](#) &&__p) noexcept
- **path** ([string_type](#) &&__source)
- template<typename _InputIterator >
__detail::_Path<_InputIterator, _InputIterator> & **append** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path<_Source> & **append** (_Source const &__source)
- template<typename _InputIterator >
__detail::_Path<_InputIterator, _InputIterator> & **assign** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path<_Source> & **assign** (_Source const &__source)
- [path](#) & **assign** ([string_type](#) &&__source)
- [iterator](#) **begin** () const noexcept
- const value_type * **c_str** () const noexcept
- void **clear** () noexcept
- int **compare** (const [basic_string_view](#)< value_type > __s) const
- int **compare** (const [path](#) &__p) const noexcept
- int **compare** (const [string_type](#) &__s) const
- int **compare** (const value_type * __s) const
- template<typename _InputIterator >
__detail::_Path<_InputIterator, _InputIterator> & **concat** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path<_Source> & **concat** (_Source const &__x)
- bool **empty** () const noexcept
- [iterator](#) **end** () const noexcept
- [path](#) **extension** () const
- [path](#) **filename** () const

- [std::string generic_string](#) () const
- template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
[std::basic_string](#)< _CharT, _Traits, _Allocator > **generic_string** (const _Allocator &__a= _Allocator()) const
- [std::u16string generic_u16string](#) () const
- [std::u32string generic_u32string](#) () const
- [std::string generic_u8string](#) () const
- [std::wstring generic_wstring](#) () const
- bool **has_extension** () const
- bool **has_filename** () const
- bool **has_parent_path** () const
- bool **has_relative_path** () const
- bool **has_root_directory** () const
- bool **has_root_name** () const
- bool **has_root_path** () const
- bool **has_stem** () const
- bool **is_absolute** () const
- bool **is_relative** () const
- [path](#) & **make_preferred** ()
- const [string_type](#) & **native** () const noexcept
- **operator string_type** () const
- template<typename _CharT >
__detail::_Path< _CharT *, _CharT * > & **operator+=** (_CharT __x)
- template<typename _Source >
__detail::_Path< _Source > & **operator+=** (_Source const &__x)
- [path](#) & **operator+=** ([basic_string_view](#)< value_type > __x)
- [path](#) & **operator+=** (const [path](#) &__x)
- [path](#) & **operator+=** (const [string_type](#) &__x)
- [path](#) & **operator+=** (const value_type *__x)
- [path](#) & **operator+=** (value_type __x)
- template<typename _Source >
__detail::_Path< _Source > & **operator/=** (_Source const &__source)
- [path](#) & **operator/=** (const [path](#) &__p)
- template<typename _Source >
__detail::_Path< _Source > & **operator=** (_Source const &__source)
- [path](#) & **operator=** (const [path](#) &__p)
- [path](#) & **operator=** ([path](#) &&__p) noexcept
- [path](#) & **operator=** ([string_type](#) &&__source)
- [path](#) **parent_path** () const
- [path](#) **relative_path** () const
- [path](#) & **remove_filename** ()
- [path](#) & **replace_extension** (const [path](#) &__replacement=[path](#)())
- [path](#) & **replace_filename** (const [path](#) &__replacement)
- [path](#) **root_directory** () const
- [path](#) **root_name** () const
- [path](#) **root_path** () const
- [path](#) **stem** () const
- [std::string string](#) () const
- template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
[std::basic_string](#)< _CharT, _Traits, _Allocator > **string** (const _Allocator &__a= _Allocator()) const
- void **swap** ([path](#) &__rhs) noexcept
- [std::u16string u16string](#) () const
- [std::u32string u32string](#) () const
- [std::string u8string](#) () const
- [std::wstring wstring](#) () const

Static Public Attributes

- static constexpr value_type **preferred_separator**

6.821.1 Detailed Description

A filesystem path.

The documentation for this class was generated from the following file:

- [experimental/bits/fs_path.h](#)

6.822 std::filesystem::path Class Reference

```
#include <fs_path.h>
```

Classes

- class [iterator](#)

Public Types

- using **const_iterator** = [iterator](#)
- enum **format** : unsigned char { **native_format** , **generic_format** , **auto_format** }
- using **string_type** = [std::basic_string](#)< value_type >
- using **value_type** = char

Public Member Functions

- template<typename _InputIterator , typename _Require = __detail::_Path2<_InputIterator>, typename _Req2 = __detail::_value_type_is_↔ is_char<_InputIterator>>>
path (_InputIterator __first, _InputIterator __last, const [locale](#) &__loc, **format**=auto_format)
- template<typename _InputIterator , typename _Require = __detail::_Path2<_InputIterator>>>
path (_InputIterator __first, _InputIterator __last, **format**=auto_format)
- template<typename _Source , typename _Require = __detail::_Path<_Source>>>
path (_Source const &__source, **format**=auto_format)
- template<typename _Source , typename _Require = __detail::_Path<_Source>, typename _Require2 = __detail::_value_type_is_↔ char<_Source>>>
path (_Source const &__src, const [locale](#) &__loc, **format**=auto_format)
- **path** (const [path](#) &__p)=default
- **path** ([path](#) &&__p) noexcept
- **path** ([string_type](#) &&__source, **format**=auto_format)
- template<typename _InputIterator >
__detail::_Path2< _InputIterator > & **append** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path< _Source > & **append** (_Source const &__source)
- template<typename _InputIterator >
__detail::_Path2< _InputIterator > & **assign** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path< _Source > & **assign** (_Source const &__source)
- [path](#) & **assign** ([string_type](#) &&__source)
- **iterator begin** () const noexcept
- const value_type * **c_str** () const noexcept
- void **clear** () noexcept
- int **compare** ([basic_string_view](#)< value_type > __s) const noexcept

- int **compare** (const [path](#) &__p) const noexcept
- int **compare** (const [string_type](#) &__s) const noexcept
- int **compare** (const value_type * __s) const noexcept
- template<typename _InputIterator >
__detail::_Path2< _InputIterator > & **concat** (_InputIterator __first, _InputIterator __last)
- template<typename _Source >
__detail::_Path< _Source > & **concat** (_Source const &__x)
- bool **empty** () const noexcept
- [iterator](#) **end** () const noexcept
- [path](#) **extension** () const
- [path](#) **filename** () const
- [std::string](#) **generic_string** () const
- template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
[std::basic_string](#)< _CharT, _Traits, _Allocator > **generic_string** (const _Allocator &__a= _Allocator()) const
- [std::u16string](#) **generic_u16string** () const
- [std::u32string](#) **generic_u32string** () const
- [std::string](#) **generic_u8string** () const
- [std::wstring](#) **generic_wstring** () const
- bool **has_extension** () const noexcept
- bool **has_filename** () const noexcept
- bool **has_parent_path** () const noexcept
- bool **has_relative_path** () const noexcept
- bool **has_root_directory** () const noexcept
- bool **has_root_name** () const noexcept
- bool **has_root_path** () const noexcept
- bool **has_stem** () const noexcept
- bool **is_absolute** () const noexcept
- bool **is_relative** () const noexcept
- [path](#) **lexically_normal** () const
- [path](#) **lexically_proximate** (const [path](#) &base) const
- [path](#) **lexically_relative** (const [path](#) &base) const
- [path](#) & **make_preferred** ()
- const [string_type](#) & **native** () const noexcept
- **operator string_type** () const
- template<typename _CharT >
__detail::_Path2< _CharT * > & **operator+=** (_CharT __x)
- template<typename _Source >
__detail::_Path< _Source > & **operator+=** (_Source const &__x)
- [path](#) & **operator+=** ([basic_string_view](#)< value_type > __x)
- [path](#) & **operator+=** (const [path](#) &__x)
- [path](#) & **operator+=** (const [string_type](#) &__x)
- [path](#) & **operator+=** (const value_type * __x)
- [path](#) & **operator+=** (value_type __x)
- template<typename _Source >
__detail::_Path< _Source > & **operator/=** (_Source const &__source)
- [path](#) & **operator/=** (const [path](#) &__p)
- template<typename _Source >
__detail::_Path< _Source > & **operator=** (_Source const &__source)
- [path](#) & **operator=** (const [path](#) &)
- [path](#) & **operator=** ([path](#) &&) noexcept
- [path](#) & **operator=** ([string_type](#) &&__source)

- `path parent_path () const`
- `path relative_path () const`
- `path & remove_filename ()`
- `path & replace_extension (const path &__replacement=path())`
- `path & replace_filename (const path &__replacement)`
- `path root_directory () const`
- `path root_name () const`
- `path root_path () const`
- `path stem () const`
- `std::string string () const`
- `template<typename _CharT, typename _Traits, typename _Allocator >
basic_string< _CharT, _Traits, _Allocator > string (const _Allocator &__a) const`
- `template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
std::basic_string< _CharT, _Traits, _Allocator > string (const _Allocator &__a=_Allocator()) const`
- `void swap (path &__rhs) noexcept`
- `std::u16string u16string () const`
- `std::u32string u32string () const`
- `std::string u8string () const`
- `std::wstring wstring () const`

Static Public Attributes

- static constexpr value_type **preferred_separator**

Friends

- `bool operator!= (const path &__lhs, const path &__rhs) noexcept`
- `path operator/ (const path &__lhs, const path &__rhs)`
- `bool operator< (const path &__lhs, const path &__rhs) noexcept`
- `template<typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const path &__p)`
- `bool operator<= (const path &__lhs, const path &__rhs) noexcept`
- `bool operator== (const path &__lhs, const path &__rhs) noexcept`
- `bool operator> (const path &__lhs, const path &__rhs) noexcept`
- `bool operator>= (const path &__lhs, const path &__rhs) noexcept`
- `template<typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, path &__p)`

Related Symbols

(Note that these are not member symbols.)

- `void swap (same_as< path > auto &__lhs, same_as< path > auto &__rhs) noexcept`
- `template<typename _InputIterator, typename _Require = __detail::_Path2<_InputIterator>, typename _CharT = __detail::_value_type<↔
_is_char_or_char8_t<_InputIterator>>
path u8path (_InputIterator __first, _InputIterator __last)`
- `template<typename _Source, typename _Require = __detail::_Path<_Source>, typename _CharT = __detail::_value_type_is_char_or<↔
_char8_t<_Source>>
path u8path (const _Source &__source)`

6.822.1 Detailed Description

A filesystem path.

6.822.2 Member Enumeration Documentation

format

enum `std::filesystem::path::format` : unsigned char

`path::format` is ignored in this implementation

6.822.3 Friends And Related Symbol Documentation

operator"!="

```
bool operator!= (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Compare paths.

operator/

```
path operator/ (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Append one path to another.

operator<

```
bool operator< (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Compare paths.

operator<<

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const path & __p ) [friend]
```

Write a path to a stream.

operator<=

```
bool operator<= (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Compare paths.

operator==

```
bool operator== (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Compare paths.

operator>

```
bool operator> (
    const path & __lhs,
    const path & __rhs ) [friend]
```


Compare paths.

operator>=

```
bool operator>= (
    const path & __lhs,
    const path & __rhs ) [friend]
```

Compare paths.

operator>>

```
template<typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    path & __p ) [friend]
```

Read a path from a stream.

The documentation for this class was generated from the following file:

- [bits/fs_path.h](#)

6.823 std::piecewise_constant_distribution<_RealType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- template<typename _InputIteratorB , typename _InputIteratorW >
piecewise_constant_distribution (_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin)
- **piecewise_constant_distribution** (const [param_type](#) &__p)
- template<typename _Func >
piecewise_constant_distribution ([initializer_list](#)< _RealType > __bl, _Func __fw)
- template<typename _Func >
piecewise_constant_distribution (size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw)
- template<typename _ForwardIterator , typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator , typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [std::vector](#)< double > [densities](#) () const
- [std::vector](#)< _RealType > [intervals](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- template<typename _UniformRandomNumberGenerator >
[result_type operator\(\)](#) (_UniformRandomNumberGenerator &__urng)

- template<typename _UniformRandomNumberGenerator >
result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)
- param_type param () const
- void param (const param_type &__param)
- void reset ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::piecewise_constant_distribution< _RealType1 > &__x)
- bool operator== (const piecewise_constant_distribution &__d1, const piecewise_constant_distribution &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::piecewise_constant_distribution< _RealType1 > &__x)

6.823.1 Detailed Description

template<typename _RealType = double>
class std::piecewise_constant_distribution<_RealType>

A piecewise_constant_distribution random number distribution.
The formula for the piecewise constant probability mass function is

6.823.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::piecewise_constant_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.823.3 Member Function Documentation

densities()

```
template<typename _RealType = double>
std::vector< double > std::piecewise_constant_distribution< _RealType >::densities ( ) const
[inline]
```

Returns a vector of the probability densities.
References [std::vector<_Tp, _Alloc>::empty\(\)](#).

intervals()

```
template<typename _RealType = double>
std::vector< _RealType > std::piecewise_constant_distribution< _RealType >::intervals ( ) const
[inline]
```

Returns a vector of the intervals.
References [std::vector<_Tp, _Alloc>::empty\(\)](#).

max()

```
template<typename _RealType = double>
result_type std::piecewise_constant_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::vector<_Tp, _Alloc>::back\(\)](#), and [std::vector<_Tp, _Alloc>::empty\(\)](#).

min()

```
template<typename _RealType = double>
result_type std::piecewise_constant_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

References `std::vector< _Tp, _Alloc >::empty()`, and `std::vector< _Tp, _Alloc >::front()`.

operator()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::piecewise_constant_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References `std::piecewise_constant_distribution< _RealType >::operator()()`.

Referenced by `std::piecewise_constant_distribution< _RealType >::operator()()`.

param() [1/2]

```
template<typename _RealType = double>
param_type std::piecewise_constant_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::piecewise_constant_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::piecewise_constant_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

6.823.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::piecewise_constant_distribution< _RealType1 > & __x ) [friend]
```

Inserts a `piecewise_constant_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>piecewise_constant_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const piecewise\_constant\_distribution< _RealType > & __d1,
    const piecewise\_constant\_distribution< _RealType > & __d2 ) [friend]
```

Return true if two piecewise constant distributions have the same parameters.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic\_istream< _CharT, _Traits > & operator>> (
    std::basic\_istream< _CharT, _Traits > & __is,
    std::piecewise\_constant\_distribution< _RealType1 > & __x ) [friend]
```

Extracts a `piecewise_constant_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>piecewise_constant_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.824 std::piecewise_construct_t Struct Reference

```
#include <stl_pair.h>
```

6.824.1 Detailed Description

Tag type for piecewise construction of `std::pair` objects.

The documentation for this struct was generated from the following file:

- [stl_pair.h](#)

6.825 std::piecewise_linear_distribution< _RealType > Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- `template<typename _InputIteratorB, typename _InputIteratorW >`
`piecewise_linear_distribution` (`_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin`)
- **`piecewise_linear_distribution`** (`const param_type &__p`)
- `template<typename _Func >`
`piecewise_linear_distribution` (`initializer_list< _RealType > __bl, _Func __fw`)
- `template<typename _Func >`
`piecewise_linear_distribution` (`size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate` (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate` (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `template<typename _UniformRandomNumberGenerator >`
`void __generate` (`result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `std::vector< double > densities` () const
- `std::vector< _RealType > intervals` () const
- `result_type max` () const
- `result_type min` () const
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator()` (`_UniformRandomNumberGenerator &__urng`)
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator()` (`_UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `param_type param` () const
- `void param` (`const param_type &__param`)
- `void reset` ()

Friends

- `template<typename _RealType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<<` (`std::basic_ostream< _CharT, _Traits > &__os, const std::piecewise_linear_distribution< _RealType1 > &__x`)
- `bool operator==` (`const piecewise_linear_distribution &__d1, const piecewise_linear_distribution &__d2`)
- `template<typename _RealType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>>` (`std::basic_istream< _CharT, _Traits > &__is, std::piecewise_linear_distribution< _RealType1 > &__x`)

6.825.1 Detailed Description

```
template<typename _RealType = double>
class std::piecewise_linear_distribution< _RealType >
```

A `piecewise_linear_distribution` random number distribution.
The formula for the piecewise linear probability mass function is

6.825.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
```

```
typedef _RealType std::piecewise_linear_distribution< _RealType >::result_type
```

The type of the range of the distribution.

6.825.3 Member Function Documentation

densities()

```
template<typename _RealType = double>
```

```
std::vector< double > std::piecewise_linear_distribution< _RealType >::densities ( ) const [inline]
```

Return a vector of the probability densities of the distribution.

References [std::vector<_Tp, _Alloc>::empty\(\)](#).

intervals()

```
template<typename _RealType = double>
```

```
std::vector< _RealType > std::piecewise_linear_distribution< _RealType >::intervals ( ) const [inline]
```

Return the intervals of the distribution.

References [std::vector<_Tp, _Alloc>::empty\(\)](#).

max()

```
template<typename _RealType = double>
```

```
result_type std::piecewise_linear_distribution< _RealType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.

References [std::vector<_Tp, _Alloc>::back\(\)](#), and [std::vector<_Tp, _Alloc>::empty\(\)](#).

min()

```
template<typename _RealType = double>
```

```
result_type std::piecewise_linear_distribution< _RealType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

References [std::vector<_Tp, _Alloc>::empty\(\)](#), and [std::vector<_Tp, _Alloc>::front\(\)](#).

operator>()()

```
template<typename _RealType = double>
```

```
template<typename _UniformRandomNumberGenerator >
```

```
result_type std::piecewise_linear_distribution< _RealType >::operator() (   
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::piecewise_linear_distribution<_RealType>::operator>\(\)\(\)](#).

Referenced by [std::piecewise_linear_distribution<_RealType>::operator>\(\)\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
```

```
param_type std::piecewise_linear_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::piecewise_linear_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::piecewise_linear_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

6.825.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::piecewise_linear_distribution< _RealType1 > & __x ) [friend]
```

Inserts a `piecewise_linear_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>piecewise_linear_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const piecewise_linear_distribution< _RealType > & __d1,
    const piecewise_linear_distribution< _RealType > & __d2 ) [friend]
```

Return true if two `piecewise_linear_distribution` have the same parameters.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::piecewise_linear_distribution< _RealType1 > & __x ) [friend]
```

Extracts a `piecewise_linear_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>piecewise_linear_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

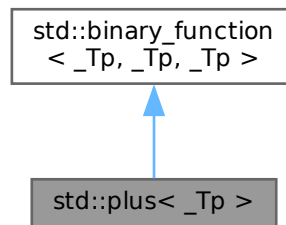
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.826 `std::plus<_Tp>` Struct Template Reference

```
#include <std_function.h>
```

Inheritance diagram for `std::plus<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- constexpr `_Tp` [operator\(\)](#) (const `_Tp` &`__x`, const `_Tp` &`__y`) const

6.826.1 Detailed Description

```
template<typename _Tp>
```

```
struct std::plus<_Tp>
```

One of the [math functors](#).

6.826.2 Member Typedef Documentation

first_argument_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `_Tp` >::`first_argument_type` [inherited]
`first_argument_type` is the type of the first argument

result_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `_Tp` >::`result_type` [inherited]
`result_type` is the return type

second_argument_type

typedef `_Tp` `std::binary_function`< `_Tp` , `_Tp` , `_Tp` >::`second_argument_type` [inherited]
`second_argument_type` is the type of the second argument

6.826.3 Member Function Documentation

operator>()()

```
template<typename _Tp >
constexpr _Tp std::plus< _Tp >::operator() (
    const _Tp & __x,
    const _Tp & __y) const [inline], [constexpr]
```

Returns the sum.

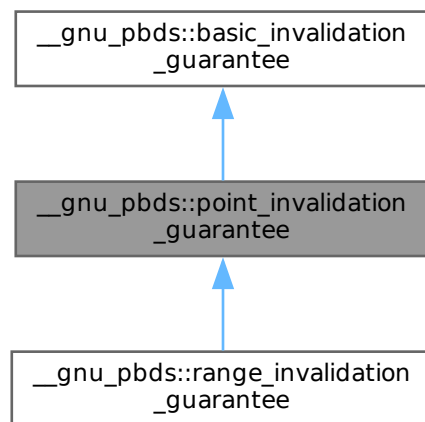
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.827 __gnu_pbds::point_invalidation_guarantee Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::point_invalidation_guarantee`:



6.827.1 Detailed Description

Signifies an invalidation guarantee that includes all those of its base, and additionally, that any point-type iterator, pointer, or reference to a container object's mapped value type is valid as long as its corresponding entry has not be erased, regardless of modifications to the container object.

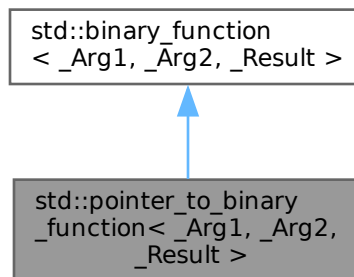
The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.828 `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >` Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >`:



Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `pointer_to_binary_function` (`_Result`(*__x)(`_Arg1`, `_Arg2`))
- `_Result operator()` (`_Arg1` __x, `_Arg2` __y) const

Protected Attributes

- `_Result`(* `_M_ptr`)(`_Arg1`, `_Arg2`)

6.828.1 Detailed Description

```
template<typename _Arg1, typename _Arg2, typename _Result>
class std::pointer_to_binary_function< _Arg1, _Arg2, _Result >
```

One of the [adaptors for function pointers](#).

6.828.2 Member Typedef Documentation

first_argument_type

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg1 std::binary_function< _Arg1, _Arg2, _Result >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

result_type

```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Result std::binary_function< _Arg1, _Arg2, _Result >::result_type [inherited]
result_type is the return type
```

second_argument_type

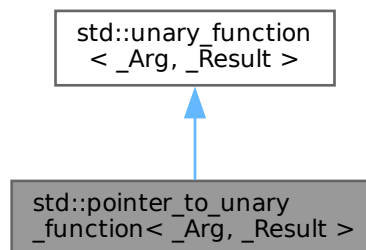
```
template<typename _Arg1 , typename _Arg2 , typename _Result >
typedef _Arg2 std::binary_function< _Arg1, _Arg2, _Result >::second_argument_type [inherited]
second_argument_type is the type of the second argument
The documentation for this class was generated from the following file:
```

- [stl_function.h](#)

6.829 std::pointer_to_unary_function< _Arg, _Result > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::pointer_to_unary_function< _Arg, _Result >:



Public Types

- typedef _Arg [argument_type](#)
- typedef _Result [result_type](#)

Public Member Functions

- [pointer_to_unary_function](#) (_Result(* __x)(_Arg))
- _Result [operator\(\)](#) (_Arg __x) const

Protected Attributes

- `_Result(*_M_ptr)(_Arg)`

6.829.1 Detailed Description

```
template<typename _Arg, typename _Result>
class std::pointer_to_unary_function<_Arg, _Result>
```

One of the [adaptors for function pointers](#).

6.829.2 Member Typedef Documentation**argument_type**

```
template<typename _Arg, typename _Result>
typedef _Arg std::unary_function<_Arg, _Result>::argument_type [inherited]
argument_type is the type of the argument
```

result_type

```
template<typename _Arg, typename _Result>
typedef _Result std::unary_function<_Arg, _Result>::result_type [inherited]
result_type is the return type
```

The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.830 `std::pointer_traits<_Ptr>` Struct Template Reference

```
#include <ptr_traits.h>
Inherits std::__ptr_traits_impl<_Ptr, _Elt>.
```

Public Types

- using [difference_type](#) = typename `__difference<_Ptr>::type`
- using [element_type](#) = `_Elt`
- using [pointer](#) = `_Ptr`
- template<typename `_Up`>
using [rebind](#) = typename `__rebind<_Ptr, _Up>::type`

Static Public Member Functions

- static pointer [pointer_to](#) (`element_type &__e`)

6.830.1 Detailed Description

```
template<typename _Ptr>
struct std::pointer_traits<_Ptr>
```

Uniform interface to all pointer-like types.

Since

C++11

6.830.2 Member Typedef Documentation

difference_type

```
template<typename _Ptr , typename _Elt >
using std::__ptr_traits_impl< _Ptr, _Elt >::difference_type = typename __difference<_Ptr>::type
[inherited]
```

The type used to represent the difference between two pointers.

element_type

```
template<typename _Ptr , typename _Elt >
using std::__ptr_traits_impl< _Ptr, _Elt >::element_type = _Elt [inherited]
```

The type pointed to.

pointer

```
template<typename _Ptr , typename _Elt >
using std::__ptr_traits_impl< _Ptr, _Elt >::pointer = _Ptr [inherited]
```

The pointer type.

rebind

```
template<typename _Ptr , typename _Elt >
template<typename _Up >
using std::__ptr_traits_impl< _Ptr, _Elt >::rebind = typename __rebind<_Ptr, _Up>::type [inherited]
```

A pointer to a different type.

6.830.3 Member Function Documentation

pointer_to()

```
template<typename _Ptr , typename _Elt , bool = is_void<_Elt>::value>
static pointer std::__ptr_traits_ptr_to< _Ptr, _Elt, bool >::pointer_to (
    element_type & __e ) [inline], [static], [inherited]
```

Obtain a pointer to an object.

Parameters

↩	A reference to an object of type element_type
↩	
↩	
↩	
<i>r</i>	

Returns

pointer::pointer_to(__e)

Precondition

pointer::pointer_to(__e) is a valid expression.

The documentation for this struct was generated from the following file:

- [ptr_traits.h](#)

6.831 `std::pointer_traits<_Tp*>` Struct Template Reference

```
#include <ptr_traits.h>
Inherits std::__ptr_traits_ptr_to<_Ptr, _Elt, bool>.
```

Public Types

- typedef `ptrdiff_t` `difference_type`
- typedef `_Tp` `element_type`
- typedef `_Tp*` `pointer`
- template<typename `_Up`>
using `rebind` = `_Up*`

Static Public Member Functions

- static `pointer` `pointer_to` (`element_type` &__e)

6.831.1 Detailed Description

```
template<typename _Tp>
struct std::pointer_traits<_Tp*>
```

Partial specialization for built-in pointers.

Since

C++11

6.831.2 Member Typedef Documentation

`difference_type`

```
template<typename _Tp>
typedef ptrdiff_t std::pointer_traits<_Tp*>::difference_type
Type used to represent the difference between two pointers.
```

`element_type`

```
template<typename _Tp>
typedef _Tp std::pointer_traits<_Tp*>::element_type
The type pointed to.
```

`pointer`

```
template<typename _Tp>
typedef _Tp* std::pointer_traits<_Tp*>::pointer
The pointer type.
```

`rebind`

```
template<typename _Tp>
template<typename _Up>
using std::pointer_traits<_Tp*>::rebind = _Up*
A pointer to a different type.
```

6.831.3 Member Function Documentation

pointer_to()

```
template<typename _Ptr , typename _Elt , bool = is_void<_Elt>::value>
static pointer std::__ptr_traits_ptr_to< _Ptr, _Elt, bool >::pointer_to (
    element_type & __e ) [inline], [static], [inherited]
```

Obtain a pointer to an object.

Parameters

↩	A reference to an object of type <code>element_type</code>
↩	
↩	
↩	
<i>r</i>	

Returns

```
pointer::pointer_to(__e)
```

Precondition

`pointer::pointer_to(__e)` is a valid expression.

The documentation for this struct was generated from the following file:

- [ptr_traits.h](#)

6.832 std::poisson_distribution<_IntType> Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **poisson_distribution** (const [param_type](#) &__p)
- **poisson_distribution** (double __mean)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- template<typename `_UniformRandomNumberGenerator` >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- double **mean** () const
- [result_type](#) **min** () const

- template<typename _UniformRandomNumberGenerator >
result_type operator() (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)
- param_type param () const
- void param (const param_type &__param)
- void reset ()

Friends

- template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::poisson_distribution< _IntType1 > &__x)
- bool operator== (const poisson_distribution &__d1, const poisson_distribution &__d2)
- template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::poisson_distribution< _IntType1 > &__x)

6.832.1 Detailed Description

template<typename _IntType = int>
class std::poisson_distribution< _IntType >

A discrete Poisson random number distribution.

The formula for the Poisson probability density function is $p(i|\mu) = \frac{\mu^i}{i!} e^{-\mu}$ where μ is the parameter of the distribution.

6.832.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::poisson_distribution< _IntType >::result_type
```

The type of the range of the distribution.

6.832.3 Member Function Documentation

max()

```
template<typename _IntType = int>
result_type std::poisson_distribution< _IntType >::max ( ) const [inline]
```

Returns the least upper bound value of the distribution.
References [std::numeric_limits< _Tp >::max\(\)](#).

mean()

```
template<typename _IntType = int>
double std::poisson_distribution< _IntType >::mean ( ) const [inline]
```

Returns the distribution parameter mean.

min()

```
template<typename _IntType = int>
result_type std::poisson_distribution< _IntType >::min ( ) const [inline]
```

Returns the greatest lower bound value of the distribution.

operator>() [1/2]

```
template<typename _IntType = int>
template<typename _UniformRandomNumberGenerator >
result_type std::poisson_distribution< _IntType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::poisson_distribution< _IntType >::operator>\(\)](#).

Referenced by [std::poisson_distribution< _IntType >::operator>\(\)](#).

operator>() [2/2]

```
template<typename _IntType >
template<typename _UniformRandomNumberGenerator >
poisson_distribution< _IntType >::result_type std::poisson_distribution< _IntType >::operator()
(
    _UniformRandomNumberGenerator & __urng,
    const param_type & __param )
```

A rejection algorithm when mean ≥ 12 and a simple method based upon the multiplication of uniform random variates otherwise. NB: The former is available only if `_GLIBCXX_USE_C99_MATH_TR1` is defined.

Reference: Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. X, Sects. 3.3 & 3.4 (+ Errata!).

References [std::abs\(\)](#), [std::numeric_limits< _Tp >::epsilon\(\)](#), [std::log\(\)](#), and [std::numeric_limits< _Tp >::max\(\)](#).

param() [1/2]

```
template<typename _IntType = int>
param_type std::poisson_distribution< _IntType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _IntType = int>
void std::poisson_distribution< _IntType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _IntType = int>
void std::poisson_distribution< _IntType >::reset ( ) [inline]
```

Resets the distribution state.

References [std::normal_distribution< _RealType >::reset\(\)](#).

6.832.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
```

```
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::poisson_distribution< _IntType1 > & __x ) [friend]
```

Inserts a poisson_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A poisson_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

operator==

```
template<typename _IntType = int>
bool operator== (
    const poisson_distribution< _IntType > & __d1,
    const poisson_distribution< _IntType > & __d2 ) [friend]
```

Return true if two Poisson distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _IntType = int>
template<typename _IntType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::poisson_distribution< _IntType1 > & __x ) [friend]
```

Extracts a poisson_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A poisson_distribution random number generator engine.

Returns

The input stream with __x extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.833 std::pmr::pool_options Struct Reference**Public Attributes**

- `size_t` [largest_required_pool_block](#)
- `size_t` [max_blocks_per_chunk](#)

6.833.1 Detailed Description

Parameters for tuning a pool resource's behaviour.

6.833.2 Member Data Documentation

max_blocks_per_chunk

`size_t std::pmr::pool_options::max_blocks_per_chunk`

Upper limit on number of blocks in a chunk.

A lower value prevents allocating huge chunks that could remain mostly unused, but means pools will need to replenished more frequently.

The documentation for this struct was generated from the following file:

- [memory_resource](#)

6.834 `__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >` Class Template Reference

`#include <priority_queue.hpp>`

Inherits `detail::container_base_dispatch::type`.

Public Types

- `typedef _Alloc allocator_type`
- `typedef Cmp_Fn cmp_fn`
- `typedef base_type::const_iterator const_iterator`
- `typedef __rebind_va::const_pointer const_pointer`
- `typedef __rebind_va::const_reference const_reference`
- `typedef Tag container_category`
- `typedef allocator_type::difference_type difference_type`
- `typedef base_type::iterator iterator`
- `typedef base_type::point_const_iterator point_const_iterator`
- `typedef base_type::point_iterator point_iterator`
- `typedef __rebind_va::pointer pointer`
- `typedef __rebind_va::reference reference`
- `typedef allocator_type::size_type size_type`
- `typedef _Tv value_type`

Public Member Functions

- [priority_queue](#) (const `cmp_fn` &`r_cmp_fn`)
- [priority_queue](#) (const [priority_queue](#) &`other`)
- `template<typename It >`
[priority_queue](#) (It `first_it`, It `last_it`)
- `template<typename It >`
[priority_queue](#) (It `first_it`, It `last_it`, const `cmp_fn` &`r_cmp_fn`)
- [priority_queue](#) & **operator=** (const [priority_queue](#) &`other`)
- void **swap** ([priority_queue](#) &`other`)

6.834.1 Detailed Description

```
template<typename _Tv, typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename  
_Alloc = std::allocator<char>>
```

```
class __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>
```

A priority queue composed of one specific heap policy.

Template Parameters

<code>_Tv</code>	Value type.
<code>Cmp_Fn</code>	Comparison functor.
<code>Tag</code>	Instantiating data structure type, see <code>container_tag</code> .
<code>_Alloc</code>	Allocator type.

Base is dispatched at compile time via `Tag`, from the following choices: `binary_heap_tag`, `binomial_heap_tag`, `pairing_heap_tag`, `rc_binomial_heap_tag`, `thin_heap_tag`

Base choices are: `detail::binary_heap`, `detail::binomial_heap`, `detail::pairing_heap`, `detail::rc_binomial_heap`, `detail::thin_heap`.

6.834.2 Constructor & Destructor Documentation

`priority_queue()` [1/3]

```
template<typename _Tv , typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename
_Alloc = std::allocator<char>>
__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (
    const cmp_fn & r_cmp_fn ) [inline]
```

Constructor taking some policy objects. `r_cmp_fn` will be copied by the `Cmp_Fn` object of the container object.

`priority_queue()` [2/3]

```
template<typename _Tv , typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename
_Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (
    It first_it,
    It last_it ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

`priority_queue()` [3/3]

```
template<typename _Tv , typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename
_Alloc = std::allocator<char>>
template<typename It >
__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (
    It first_it,
    It last_it,
    const cmp_fn & r_cmp_fn ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_cmp_fn` will be copied by the `cmp_fn` object of the container object.

The documentation for this class was generated from the following file:

- [priority_queue.hpp](#)

6.835 `std::priority_queue< _Tp, _Sequence, _Compare >` Class Template Reference

```
#include <stl_queue.h>
```

Public Types

- `typedef _Sequence::const_reference` **const_reference**

- typedef _Sequence **container_type**
- typedef _Sequence::reference **reference**
- typedef _Sequence::size_type **size_type**
- typedef _Compare **value_compare**
- typedef _Sequence::value_type **value_type**

Public Member Functions

- template<typename _Seq = _Sequence, typename _Requires = typename enable_if<__and<_is_default_constructible<_Compare>, is_↵
_default_constructible<_Seq>>::value>::type>
[priority_queue](#) ()
- template<typename _InputIterator, typename _Alloc, typename = std::RequireInputIter<_InputIterator>, typename _Requires = _↵
Uses<_Alloc>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Alloc &__alloc)
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, _Sequence &&__s)
- template<typename _InputIterator, typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, _Sequence &&__s, const
_Alloc &__alloc)
- template<typename _InputIterator, typename _Alloc, typename = std::RequireInputIter<_InputIterator>, typename _Requires = _↵
Uses<_Alloc>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, const _Alloc &__alloc)
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, const _Sequence &__s)
- template<typename _InputIterator, typename _Alloc, typename = std::RequireInputIter<_InputIterator>, typename _Requires = _↵
Uses<_Alloc>>>
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, const _Sequence &__s, const
_Alloc &__alloc)
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
[priority_queue](#) (_InputIterator __first, _InputIterator __last, const _Compare &__x=_Compare())
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (const _Compare &__x, _Sequence &&__c, const _Alloc &__a)
- **priority_queue** (const _Compare &__x, _Sequence &&__s=_Sequence())
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (const _Compare &__x, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (const _Compare &__x, const _Sequence &__c, const _Alloc &__a)
- **priority_queue** (const _Compare &__x, const _Sequence &__s)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue (const [priority_queue](#) &__q, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>>
priority_queue ([priority_queue](#) &&__q, const _Alloc &__a)
- template<typename... _Args>
void **emplace** (_Args &&... __args)
- bool [empty](#) () const
- void [pop](#) ()
- void [push](#) (const value_type &__x)
- void [push](#) (value_type &&__x)
- size_type [size](#) () const
- void **swap** ([priority_queue](#) &__pq) noexcept(__and<_is_nothrow_swappable<_↵
Sequence>, _is_nothrow_swappable<_Compare>>::value)
- const_reference [top](#) () const

Protected Attributes

- `_Sequence c`
- `_Compare comp`

6.835.1 Detailed Description

```
template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>>
class std::priority_queue< _Tp, _Sequence, _Compare >
```

A standard container automatically sorting its contents.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Sequence</code>	Type of underlying sequence, defaults to <code>vector<_Tp></code> .
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Sequence::value_type></code> .

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces priority-based sorting and queue behavior. Very few of the standard container/sequence interface requirements are met (e.g., iterators).

The second template parameter defines the type of the underlying sequence/container. It defaults to `std::vector`, but it can be any type that supports `front()`, `push_back`, `pop_back`, and random-access iterators, such as `std::deque` or an appropriate user-defined type.

The third template parameter supplies the means of making priority comparisons. It defaults to `less<value_type>` but can be anything defining a strict weak ordering.

Members not found in *normal* containers are `container_type`, which is a typedef for the second `Sequence` parameter, and `push`, `pop`, and `top`, which are standard queue operations.

Note

No equality/comparison operators are provided for `priority_queue`.

Sorting of the elements takes place as they are added to, and removed from, the `priority_queue` using the `priority_queue`'s member functions. If you access the elements by other means, and change their data such that the sorting order would be different, the `priority_queue` will not re-sort the elements for you. (How could it know to do so?)

6.835.2 Constructor & Destructor Documentation

`priority_queue()` [1/2]

```
template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>>
```

```
template<typename _Seq = _Sequence, typename _Requires = typename enable_if<__and<is_default_constructible<_Compare>, is_default_constructible<_Seq>>::value>::type>
```

```
std::priority_queue< _Tp, _Sequence, _Compare >::priority_queue ( ) [inline]
```

Default constructor creates no elements.

`priority_queue()` [2/2]

```
template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>>
```

```
template<typename _InputIterator , typename = std::RequireInputIter<_InputIterator>>
```

```
std::priority_queue< _Tp, _Sequence, _Compare >::priority_queue (
    _InputIterator __first,
```

```

    __InputIterator __last,
    const _Compare & __x = _Compare() ) [inline]

```

Builds a queue from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__x</code>	A comparison functor describing a strict weak ordering.
<code>__s</code>	An initial sequence with which to start.

Begins by copying `__s`, inserting a copy of the elements from `[first,last)` into the copy of `__s`, then ordering the copy according to `__x`.

For more information on function objects, see the documentation on [functor base classes](#).

6.835.3 Member Function Documentation

empty()

```

template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _↔
Sequence::value_type>>

```

```

bool std::priority_queue<_Tp, _Sequence, _Compare >::empty ( ) const [inline]

```

Returns true if the queue is empty.

Referenced by [__gnu_parallel::multiseq_partition\(\)](#), and [__gnu_parallel::multiseq_selection\(\)](#).

pop()

```

template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _↔
Sequence::value_type>>

```

```

void std::priority_queue<_Tp, _Sequence, _Compare >::pop ( ) [inline]

```

Removes first element.

This is a typical queue operation. It shrinks the queue by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop()` is called.

Referenced by [__gnu_parallel::multiseq_partition\(\)](#), and [__gnu_parallel::multiseq_selection\(\)](#).

push()

```

template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _↔
Sequence::value_type>>

```

```

void std::priority_queue<_Tp, _Sequence, _Compare >::push (
    const value_type & __x ) [inline]

```

Add data to the queue.

Parameters

<code>__↔</code>	Data to be added.
<code>__x</code>	

This is a typical queue operation. The time complexity of the operation depends on the underlying sequence.

Referenced by [__gnu_parallel::multiseq_partition\(\)](#), and [__gnu_parallel::multiseq_selection\(\)](#).

size()

```
template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Tp>
Sequence::value_type>>
```

```
size_type std::priority_queue< _Tp, _Sequence, _Compare >::size ( ) const [inline]
```

Returns the number of elements in the queue.

top()

```
template<typename _Tp , typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Tp>
Sequence::value_type>>
```

```
const_reference std::priority_queue< _Tp, _Sequence, _Compare >::top ( ) const [inline]
```

Returns a read-only (constant) reference to the data at the first element of the queue.

Referenced by [__gnu_parallel::multiseq_partition\(\)](#), and [__gnu_parallel::multiseq_selection\(\)](#).

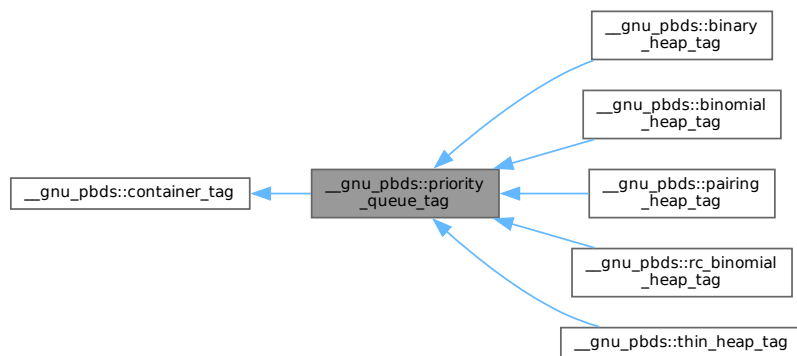
The documentation for this class was generated from the following file:

- [stl_queue.h](#)

6.836 __gnu_pbds::priority_queue_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::priority_queue_tag`:

**6.836.1 Detailed Description**

Basic priority-queue.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.837 __gnu_pbds::detail::probe_fn_base<_Alloc> Class Template Reference

```
#include <probe_fn_base.hpp>
```

6.837.1 Detailed Description

```
template<typename _Alloc>
class __gnu_pbds::detail::probe_fn_base<_Alloc>
```

Probe functor base.

The documentation for this class was generated from the following file:

- [probe_fn_base.hpp](#)

6.838 `__gnu_cxx::project1st<_Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::Project1st<_Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Arg1` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Arg1 operator()` (`const _Arg1 &__x, const _Arg2 &`) `const`

6.838.1 Detailed Description

```
template<class _Arg1, class _Arg2>
struct __gnu_cxx::project1st<_Arg1, _Arg2>
```

An [SGI extension](#) .

6.838.2 Member Typedef Documentation

`first_argument_type`

```
typedef _Arg1 std::binary_function<_Arg1, _Arg2, _Arg1>::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

`result_type`

```
typedef _Arg1 std::binary_function<_Arg1, _Arg2, _Arg1>::result_type [inherited]
result_type is the return type
```

`second_argument_type`

```
typedef _Arg2 std::binary_function<_Arg1, _Arg2, _Arg1>::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.839 `__gnu_cxx::project2nd<_Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::Project2nd<_Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Arg2` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Arg2 operator()` (`const _Arg1 &`, `const _Arg2 &__y`) `const`

6.839.1 Detailed Description

```
template<class _Arg1, class _Arg2>
struct __gnu_cxx::project2nd< _Arg1, _Arg2 >
```

An [SGI extension](#) .

6.839.2 Member Typedef Documentation

`first_argument_type`

```
typedef _Arg1 std::binary\_function< _Arg1, _Arg2, _Arg2 >::first_argument_type [inherited]
first_argument_type is the type of the first argument
```

`result_type`

```
typedef _Arg2 std::binary\_function< _Arg1, _Arg2, _Arg2 >::result_type [inherited]
result_type is the return type
```

`second_argument_type`

```
typedef _Arg2 std::binary\_function< _Arg1, _Arg2, _Arg2 >::second_argument_type [inherited]
second_argument_type is the type of the second argument
```

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.840 `std::projected< _Iter, _Proj >` Struct Template Reference

```
#include <iterator_concepts.h>
```

Public Types

- using `value_type` = [remove_cvref_t](#)< `indirect_result_t`< `_Proj &`, `_Iter` > >

Public Member Functions

- `indirect_result_t`< `_Proj &`, `_Iter` > `operator*` () `const`

6.840.1 Detailed Description

```
template<indirectly_readable _Iter, indirectly_regular_unary_invocable< _Iter > _Proj>
struct std::projected< _Iter, _Proj >
```

[projected], projected

The documentation for this struct was generated from the following file:

- [iterator_concepts.h](#)

6.841 std::promise< _Res > Class Template Reference

Public Member Functions

- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &`, `promise &&__rhs`)
- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &__a`)
- `promise` (`const promise &`)=`delete`
- `promise` (`promise &&__rhs`) `noexcept`
- `future< _Res >` `get_future` ()
- `promise & operator=` (`const promise &`)=`delete`
- `promise & operator=` (`promise &&__rhs`) `noexcept`
- `void set_exception` (`exception_ptr __p`)
- `void set_exception_at_thread_exit` (`exception_ptr __p`)
- `void set_value` (`_Res &&__r`)
- `void set_value` (`const _Res &__r`)
- `void set_value_at_thread_exit` (`_Res &&__r`)
- `void set_value_at_thread_exit` (`const _Res &__r`)
- `void swap` (`promise &__rhs`) `noexcept`

Friends

- `template<typename , typename >`
`struct _State::Setter`

6.841.1 Detailed Description

`template<typename _Res>`
`class std::promise< _Res >`

Primary template for promise.

The documentation for this class was generated from the following file:

- [future](#)

6.842 std::promise< _Res & > Class Template Reference

Public Member Functions

- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &`, `promise &&__rhs`)
- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &__a`)
- `promise` (`const promise &`)=`delete`
- `promise` (`promise &&__rhs`) `noexcept`
- `future< _Res & >` `get_future` ()
- `promise & operator=` (`const promise &`)=`delete`
- `promise & operator=` (`promise &&__rhs`) `noexcept`
- `void set_exception` (`exception_ptr __p`)
- `void set_exception_at_thread_exit` (`exception_ptr __p`)
- `void set_value` (`_Res &__r`)
- `void set_value_at_thread_exit` (`_Res &__r`)
- `void swap` (`promise &__rhs`) `noexcept`

Friends

- `template<typename , typename >`
`struct _State::_Setter`

6.842.1 Detailed Description

`template<typename _Res>`
`class std::promise<_Res & >`

Partial specialization for `promise<R&>`

The documentation for this class was generated from the following file:

- [future](#)

6.843 std::promise< void > Class Reference

Public Member Functions

- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &`, [promise](#) &&__rhs)
- `template<typename _Allocator >`
`promise` (`allocator_arg_t`, `const _Allocator &`__a)
- `promise` (`const promise &`)=delete
- `promise` ([promise](#) &&__rhs) noexcept
- `future< void > get_future` ()
- `promise & operator=` (`const promise &`)=delete
- `promise & operator=` ([promise](#) &&__rhs) noexcept
- `void set_exception` ([exception_ptr](#) __p)
- `void set_exception_at_thread_exit` ([exception_ptr](#) __p)
- `void set_value` ()
- `void set_value_at_thread_exit` ()
- `void swap` ([promise](#) &__rhs) noexcept

Friends

- `template<typename , typename >`
`struct _State::_Setter`

6.843.1 Detailed Description

Explicit specialization for `promise<void>`

The documentation for this class was generated from the following file:

- [future](#)

6.844 std::experimental::fundamentals_v2::propagate_const< _Tp > Class Template Reference

Public Types

- `typedef remove_reference_t< decltype(*std::declval<_Tp & >())> element_type`

Public Member Functions

- template<typename _Up, typename [enable_if](#)<__and<[is_constructible](#)<_Tp, _Up &&>, [is_convertible](#)<_Up &&, _Tp>, __not<__is_propagate_const<typename [decay](#)<_Up>::type>>>::value, bool>::type = true>
constexpr **propagate_const** (_Up &&__u)
- template<typename _Up, typename [enable_if](#)<__and<[is_constructible](#)<_Tp, _Up &&>, __not<[is_convertible](#)<_Up &&, _Tp>>, __not<__is_propagate_const<typename [decay](#)<_Up>::type>>>::value, bool>::type = false>
constexpr **propagate_const** (_Up &&__u)
- **propagate_const** (const [propagate_const](#) &__p)=delete
- constexpr **propagate_const** ([propagate_const](#) &&__p)=default
- template<typename _Up, typename [enable_if](#)<__and<[is_constructible](#)<_Tp, _Up &&>, [is_convertible](#)<_Up &&, _Tp>>>::value, bool>::type = true>
constexpr **propagate_const** ([propagate_const](#)<_Up> &&__pu)
- template<typename _Up, typename [enable_if](#)<__and<[is_constructible](#)<_Tp, _Up &&>, __not<[is_convertible](#)<_Up &&, _Tp>>>::value, bool>::type = false>
constexpr **propagate_const** ([propagate_const](#)<_Up> &&__pu)
- constexpr element_type * **get** ()
- constexpr const element_type * **get** () const
- constexpr **operator bool** () const
- template<typename _Up = _Tp, typename [enable_if](#)<__or<[is_pointer](#)<_Up>, [is_convertible](#)<_Up, const element_type * >>>::value, bool>::type = true>
constexpr **operator const element_type *** () const
- template<typename _Up = _Tp, typename [enable_if](#)<__or<[is_pointer](#)<_Up>, [is_convertible](#)<_Up, const element_type * >>>::value, bool>::type = true>
constexpr **operator element_type *** ()
- constexpr element_type & **operator*** ()
- constexpr const element_type & **operator*** () const
- constexpr element_type * **operator->** ()
- constexpr const element_type * **operator->** () const
- template<typename _Up, typename = typename [enable_if](#)<__and<[is_convertible](#)<_Up&&, _Tp>, __not<__is_propagate_const<typename [decay](#)<_Up>::type>>>::value>::type>
constexpr [propagate_const](#) & **operator=** (_Up &&__u)
- [propagate_const](#) & **operator=** (const [propagate_const](#) &__p)=delete
- constexpr [propagate_const](#) & **operator=** ([propagate_const](#) &&__p)=default
- template<typename _Up, typename = typename [enable_if](#)<[is_convertible](#)<_Up&&, _Tp>::value>::type>
constexpr [propagate_const](#) & **operator=** ([propagate_const](#)<_Up> &&__pu)
- constexpr void **swap** ([propagate_const](#) &__pt) noexcept(__is_nothrow_swappable<_Tp>::value)

Friends

- template<typename _Up>
constexpr const _Up & **get_underlying** (const [propagate_const](#)<_Up> &__pt) noexcept
- template<typename _Up>
constexpr _Up & **get_underlying** ([propagate_const](#)<_Up> &__pt) noexcept

6.844.1 Detailed Description

template<typename _Tp>
class std::experimental::fundamentals_v2::propagate_const<_Tp>

Const-propagating wrapper.

The documentation for this class was generated from the following file:

- [propagate_const](#)

6.845 `__gnu_pbds::quadratic_probe_fn< Size_Type >` Class Template Reference

```
#include <hash_policy.hpp>
```

Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- void **swap** (`quadratic_probe_fn< Size_Type > &other`)

Protected Member Functions

- `size_type` **operator()** (`size_type i`) const

6.845.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::quadratic_probe_fn< Size_Type >
```

A probe sequence policy using square increments.

6.845.2 Member Function Documentation

operator()()

```
template<typename Size_Type = std::size_t>
size_type __gnu_pbds::quadratic_probe_fn< Size_Type >::operator() (
    size_type i ) const [inline], [protected]
```

Returns the i-th offset from the hash value.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

6.846 `std::queue< _Tp, _Sequence >` Class Template Reference

```
#include <stl_queue.h>
```

Public Types

- typedef `_Sequence::const_reference` **const_reference**
- typedef `_Sequence` **container_type**
- typedef `_Sequence::reference` **reference**
- typedef `_Sequence::size_type` **size_type**
- typedef `_Sequence::value_type` **value_type**

Public Member Functions

- template<typename `_Seq` = `_Sequence`, typename `_Requires` = `typename enable_if<is_default_constructible<_Seq>::value>::type>`
`queue` ()
- **queue** (`_Sequence &&__c`)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
queue (`_Sequence &&__c`, const `_Alloc &__a`)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
queue (const `_Alloc &__a`)

- **queue** (const _Sequence &__c)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
queue (const _Sequence &__c, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
queue (const **queue** &__q, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
queue (**queue** &&__q, const _Alloc &__a)
- reference **back** ()
- const_reference **back** () const
- template<typename... _Args>
decltype(auto) **emplace** (_Args &&... __args)
- bool **empty** () const
- reference **front** ()
- const_reference **front** () const
- void **pop** ()
- void **push** (const value_type &__x)
- void **push** (value_type &&__x)
- size_type **size** () const
- void **swap** (**queue** &__q) noexcept(__is_nothrow_swappable< _Sequence >::value)

Protected Attributes

- _Sequence **c**

Friends

- template<typename _Tp1, typename _Seq1 >
bool **operator**< (const **queue**< _Tp1, _Seq1 > &, const **queue**< _Tp1, _Seq1 > &)
- template<typename _Tp1, typename _Seq1 >
bool **operator**== (const **queue**< _Tp1, _Seq1 > &, const **queue**< _Tp1, _Seq1 > &)

6.846.1 Detailed Description

template<typename _Tp, typename _Sequence = deque<_Tp>>
class std::queue< _Tp, _Sequence >

A standard container giving FIFO behavior.

Template Parameters

_Tp	Type of element.
_Sequence	Type of underlying sequence, defaults to deque<_Tp>.

Meets many of the requirements of a **container**, but does not define anything to do with iterators. Very few of the other standard container interfaces are defined.

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces strict first-in-first-out queue behavior.

The second template parameter defines the type of the underlying sequence/container. It defaults to std::deque, but it can be any type that supports **front**, **back**, **push_back**, and **pop_front**, such as std::list or an appropriate user-defined type.

Members not found in *normal* containers are **container_type**, which is a typedef for the second Sequence parameter, and **push** and **pop**, which are standard queue/FIFO operations.

6.846.2 Constructor & Destructor Documentation

queue()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
template<typename _Seq = _Sequence, typename _Requires = typename enable_if<is_default_constructible<↵
_Seq>::value>::type>
```

```
std::queue< _Tp, _Sequence >::queue ( ) [inline]
```

Default constructor creates no elements.

6.846.3 Member Function Documentation

back() [1/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
reference std::queue< _Tp, _Sequence >::back ( ) [inline]
```

Returns a read/write reference to the data at the last element of the queue.

References [std::queue< _Tp, _Sequence >::c](#).

back() [2/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
const_reference std::queue< _Tp, _Sequence >::back ( ) const [inline]
```

Returns a read-only (constant) reference to the data at the last element of the queue.

References [std::queue< _Tp, _Sequence >::c](#).

empty()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
bool std::queue< _Tp, _Sequence >::empty ( ) const [inline]
```

Returns true if the queue is empty.

References [std::queue< _Tp, _Sequence >::c](#).

front() [1/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
reference std::queue< _Tp, _Sequence >::front ( ) [inline]
```

Returns a read/write reference to the data at the first element of the queue.

References [std::queue< _Tp, _Sequence >::c](#).

front() [2/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
const_reference std::queue< _Tp, _Sequence >::front ( ) const [inline]
```

Returns a read-only (constant) reference to the data at the first element of the queue.

References [std::queue< _Tp, _Sequence >::c](#).

pop()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
```

```
void std::queue< _Tp, _Sequence >::pop ( ) [inline]
```

Removes first element.

This is a typical queue operation. It shrinks the queue by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before pop() is called.

References [std::queue< _Tp, _Sequence >::c](#).

push()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
void std::queue< _Tp, _Sequence >::push (
    const value_type & __x ) [inline]
```

Add data to the end of the queue.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical queue operation. The function creates an element at the end of the queue and assigns the given data to it. The time complexity of the operation depends on the underlying sequence.

References [std::queue< _Tp, _Sequence >::c](#).

size()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
size_type std::queue< _Tp, _Sequence >::size ( ) const [inline]
```

Returns the number of elements in the queue.

References [std::queue< _Tp, _Sequence >::c](#).

6.846.4 Member Data Documentation**c**

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
_Sequence std::queue< _Tp, _Sequence >::c [protected]
```

c is the underlying container.

Referenced by [std::queue< _Tp, _Sequence >::back\(\)](#), [std::queue< _Tp, _Sequence >::back\(\)](#), [std::queue< _Tp, _Sequence >::empty](#), [std::queue< _Tp, _Sequence >::front\(\)](#), [std::queue< _Tp, _Sequence >::front\(\)](#), [std::operator<\(\)](#), [std::operator==\(\(\)\)](#), [std::queue< _Tp, _Sequence >::pop\(\)](#), [std::queue< _Tp, _Sequence >::push\(\)](#), and [std::queue< _Tp, _Sequence >::size\(\)](#).

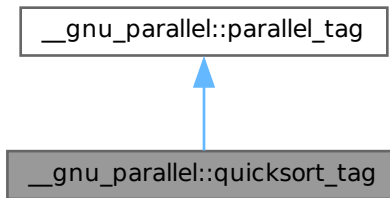
The documentation for this class was generated from the following file:

- [stl_queue.h](#)

6.847 __gnu_parallel::quicksort_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for `__gnu_parallel::quicksort_tag`:



Public Member Functions

- `quicksort_tag` (`_ThreadIndex` __num_threads)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` __num_threads)

6.847.1 Detailed Description

Forces parallel sorting using unbalanced quicksort at compile time.

6.847.2 Member Function Documentation

`__get_num_threads()`

`_ThreadIndex` `__gnu_parallel::parallel_tag::__get_num_threads` () [inline], [inherited]

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

void `__gnu_parallel::parallel_tag::set_num_threads` (
 `_ThreadIndex` __num_threads) [inline], [inherited]

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

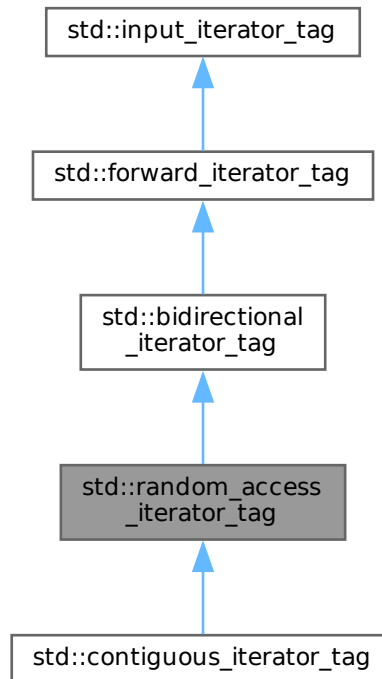
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.848 std::random_access_iterator_tag Struct Reference

```
#include <stl_iterator_base_types.h>
```

Inheritance diagram for std::random_access_iterator_tag:



6.848.1 Detailed Description

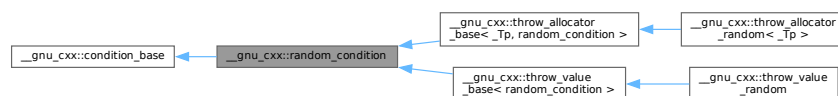
Random-access iterators support a superset of bidirectional iterator operations. The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

6.849 __gnu_cxx::random_condition Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for __gnu_cxx::random_condition:



Classes

- struct [always_adjustor](#)
- struct [group_adjustor](#)
- struct [never_adjustor](#)

Public Member Functions

- void **seed** (unsigned long __s)

Static Public Member Functions

- static void **set_probability** (double __p)
- static void **throw_conditionally** ()

6.849.1 Detailed Description

Base class for random probability control and throw.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.850 std::random_device Class Reference

```
#include <random.h>
```

Public Types

- typedef unsigned int [result_type](#)

Public Member Functions

- **random_device** (const [random_device](#) &)=delete
- **random_device** (const [std::string](#) &__token)
- double **entropy** () const noexcept
- [result_type](#) **operator**() ()
- void **operator=** (const [random_device](#) &)=delete

Static Public Member Functions

- static constexpr [result_type](#) **max** ()
- static constexpr [result_type](#) **min** ()

6.850.1 Detailed Description

A standard interface to a platform-specific non-deterministic random number generator (if any are available).

6.850.2 Member Typedef Documentation

result_type

```
typedef unsigned int std::random\_device::result\_type
```

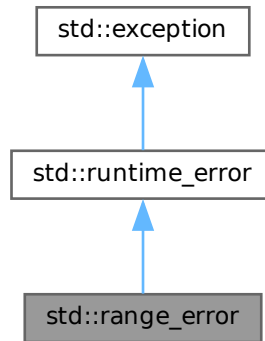
The type of the generated random value.

The documentation for this class was generated from the following file:

- [random.h](#)

6.851 `std::range_error` Class Reference

Inheritance diagram for `std::range_error`:



Public Member Functions

- `range_error` (const char *) `_GLIBCXX_TXN_SAFE`
- `range_error` (const [range_error](#) &)=default
- `range_error` (const [string](#) &__arg) `_GLIBCXX_TXN_SAFE`
- `range_error` ([range_error](#) &&)=default
- [range_error](#) & `operator=` (const [range_error](#) &)=default
- [range_error](#) & `operator=` ([range_error](#) &&)=default
- virtual const char * `what` () const noexcept

6.851.1 Detailed Description

Thrown to indicate range errors in internal computations.

6.851.2 Member Function Documentation

`what()`

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

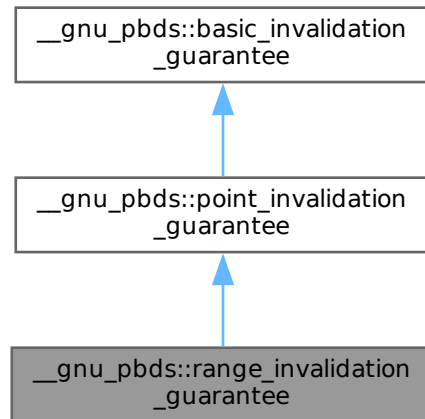
The documentation for this class was generated from the following file:

- [stdexcept](#)

6.852 `__gnu_pbds::range_invalidation_guarantee` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::range_invalidation_guarantee`:



6.852.1 Detailed Description

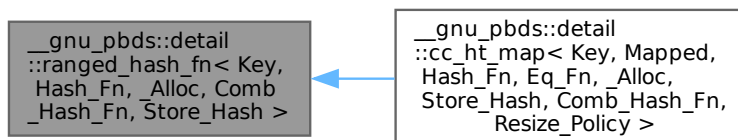
Signifies an invalidation guarantee that includes all those of its base, and additionally, that any range-type iterator (including the returns of `begin()` and `end()`) is in the correct relative positions to other range-type iterators as long as its corresponding entry has not be erased, regardless of modifications to the container object.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.853 `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash >`:



6.853.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn, bool Store_Hash>
class __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash >
```

Primary template.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

6.854 `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >` Class Template Reference

```
#include <ranged_hash_fn.hpp>
```

Inherits `Hash_Fn`, and `Comb_Hash_Fn`.

Protected Types

- typedef `Comb_Hash_Fn` **comb_hash_fn_base**
- typedef `Hash_Fn` **hash_fn_base**
- typedef [rebind_traits](#)< `_Alloc`, `Key` >::const_reference **key_const_reference**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_hash_fn** (`size_type`)
- **ranged_hash_fn** (`size_type`, const `Hash_Fn` &)
- **ranged_hash_fn** (`size_type`, const `Hash_Fn` &, const `Comb_Hash_Fn` &)
- void **notify_resized** (`size_type`)
- `size_type` **operator()** (`key_const_reference`) const
- void **swap** ([ranged_hash_fn](#)< `Key`, `Hash_Fn`, `_Alloc`, `Comb_Hash_Fn`, `false` > &)

6.854.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >
```

Specialization 1 The client supplies a hash function and a ranged hash function, and requests that hash values not be stored.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

6.855 `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >` Class Template Reference

```
#include <ranged_hash_fn.hpp>
```

Inherits `Hash_Fn`, and `Comb_Hash_Fn`.

Protected Types

- typedef `Comb_Hash_Fn` **comb_hash_fn_base**
- typedef [std::pair](#)< `size_type`, `size_type` > **comp_hash**
- typedef `Hash_Fn` **hash_fn_base**
- typedef [rebind_traits](#)< `_Alloc`, `Key` >::const_reference **key_const_reference**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Hash_Fn &)
- **ranged_hash_fn** (size_type, const Hash_Fn &, const Comb_Hash_Fn &)
- void **notify_resized** (size_type)
- **comp_hash operator()** (key_const_reference) const
- **comp_hash operator()** (key_const_reference, size_type) const
- void **swap** (ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true > &)

6.855.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >
```

Specialization 2 The client supplies a hash function and a ranged hash function, and requests that hash values be stored.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

6.856 **__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false >** > Class Template Reference

```
#include <ranged_hash_fn.hpp>
Inherits Comb_Hash_Fn.
```

Protected Types

- typedef Comb_Hash_Fn **comb_hash_fn_base**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Comb_Hash_Fn &)
- **ranged_hash_fn** (size_type, const [null_type](#) &, const Comb_Hash_Fn &)
- void **swap** (ranged_hash_fn< Key, [null_type](#), _Alloc, Comb_Hash_Fn, false > &)

6.856.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false >
```

Specialization 3 The client does not supply a hash function (by specifying null_type as the Hash_Fn parameter), and requests that hash values not be stored.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

6.857 **__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true >** > Class Template Reference

```
#include <ranged_hash_fn.hpp>
Inherits Comb_Hash_Fn.
```

Protected Types

- typedef Comb_Hash_Fn **comb_hash_fn_base**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Comb_Hash_Fn &)
- **ranged_hash_fn** (size_type, const [null_type](#) &, const Comb_Hash_Fn &)
- void **swap** ([ranged_hash_fn](#)< Key, [null_type](#), _Alloc, Comb_Hash_Fn, true > &)

6.857.1 Detailed Description

template<typename Key, typename _Alloc, typename Comb_Hash_Fn>
class `__gnu_pbds::detail::ranged_hash_fn`< Key, [null_type](#), _Alloc, Comb_Hash_Fn, true >

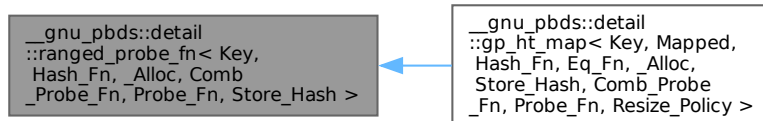
Specialization 4 The client does not supply a hash function (by specifying `null_type` as the Hash_Fn parameter), and requests that hash values be stored.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

6.858 `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ranged_probe_fn`< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >:



6.858.1 Detailed Description

template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn, bool Store_Hash>
class `__gnu_pbds::detail::ranged_probe_fn`< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >

Primary template.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

6.859 `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >` Class Template Reference

#include <ranged_probe_fn.hpp>

Inherits Hash_Fn, Comb_Probe_Fn, and Probe_Fn.

Protected Types

- typedef Comb_Probe_Fn **comb_probe_fn_base**
- typedef Hash_Fn **hash_fn_base**
- typedef [rebind_traits](#)< _Alloc, Key >::const_reference **key_const_reference**
- typedef Probe_Fn **probe_fn_base**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_probe_fn** (size_type)
- **ranged_probe_fn** (size_type, const Hash_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &, const Probe_Fn &)
- void **notify_resized** (size_type)
- size_type **operator()** (key_const_reference) const
- size_type **operator()** (key_const_reference, size_type, size_type) const
- void **swap** ([ranged_probe_fn](#)< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false > &)

6.859.1 Detailed Description

template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn>

class **__gnu_pbds::detail::ranged_probe_fn**< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >

Specialization 1

The client supplies a probe function and a ranged probe function, and requests that hash values not be stored.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

6.860 **__gnu_pbds::detail::ranged_probe_fn**< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true > Class Template Reference

#include <ranged_probe_fn.hpp>

Inherits Hash_Fn, Comb_Probe_Fn, and Probe_Fn.

Protected Types

- typedef Comb_Probe_Fn **comb_probe_fn_base**
- typedef [std::pair](#)< size_type, size_type > **comp_hash**
- typedef Hash_Fn **hash_fn_base**
- typedef [rebind_traits](#)< _Alloc, Key >::const_reference **key_const_reference**
- typedef Probe_Fn **probe_fn_base**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_probe_fn** (size_type)
- **ranged_probe_fn** (size_type, const Hash_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &, const Probe_Fn &)
- void **notify_resized** (size_type)
- [comp_hash](#) **operator()** (key_const_reference) const

- `size_type operator()` (`key_const_reference`, `size_type`) `const`
- `size_type operator()` (`key_const_reference`, `size_type`, `size_type`) `const`
- `void swap` (`ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true > &`)

6.860.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn>
class __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >
```

Specialization 2- The client supplies a probe function and a ranged probe function, and requests that hash values not be stored.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

6.861 `__gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >` Class Template Reference

```
#include <ranged_probe_fn.hpp>
```

Inherits `Comb_Probe_Fn`.

Protected Types

- `typedef Comb_Probe_Fn comb_probe_fn_base`
- `typedef rebind_traits< _Alloc, Key >::const_reference key_const_reference`
- `typedef _Alloc::size_type size_type`

Protected Member Functions

- `ranged_probe_fn` (`size_type size`)
- `ranged_probe_fn` (`size_type`, `const Comb_Probe_Fn &r_comb_probe_fn`)
- `ranged_probe_fn` (`size_type`, `const null_type &`, `const Comb_Probe_Fn &r_comb_probe_fn`, `const null_type &`)
- `void swap` (`ranged_probe_fn &other`)

6.861.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Comb_Probe_Fn>
class __gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >
```

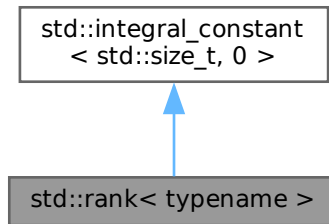
Specialization 3 and 4 The client does not supply a hash function or probe function, and requests that hash values not be stored.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

6.862 `std::rank< typename >` Struct Template Reference

Inheritance diagram for `std::rank< typename >`:



Public Types

- typedef [integral_constant](#)< `std::size_t`, `__v` > **type**
- typedef `std::size_t` **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `std::size_t` **value**

6.862.1 Detailed Description

```
template<typename>
struct std::rank< typename >
```

rank

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.863 `std::ratio< _Num, _Den >` Struct Template Reference

Public Types

- typedef [ratio](#)< `num`, `den` > **type**

Static Public Attributes

- static constexpr `intmax_t` **den**
- static constexpr `intmax_t` **num**

6.863.1 Detailed Description

```
template<intmax_t _Num, intmax_t _Den = 1>
struct std::ratio< _Num, _Den >
```

Provides compile-time rational arithmetic.

This class template represents any finite rational number with a numerator and denominator representable by compile-time constants of type intmax_t. The ratio is simplified when instantiated.

For example:

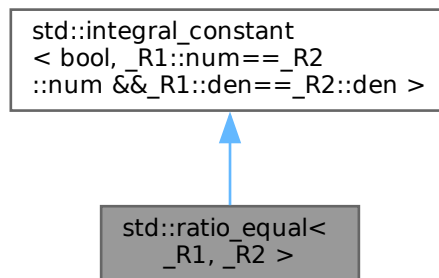
```
std::ratio<7,-21>::num == -1;
std::ratio<7,-21>::den == 3;
```

The documentation for this struct was generated from the following file:

- [ratio](#)

6.864 std::ratio_equal< _R1, _R2 > Struct Template Reference

Inheritance diagram for std::ratio_equal< _R1, _R2 >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.864.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_equal< _R1, _R2 >
```

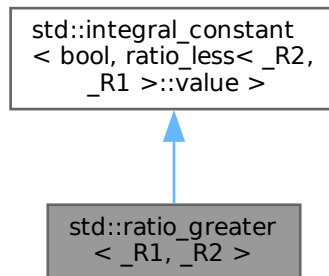
ratio_equal

The documentation for this struct was generated from the following file:

- [ratio](#)

6.865 std::ratio_greater< _R1, _R2 > Struct Template Reference

Inheritance diagram for std::ratio_greater< _R1, _R2 >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.865.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_greater< _R1, _R2 >
```

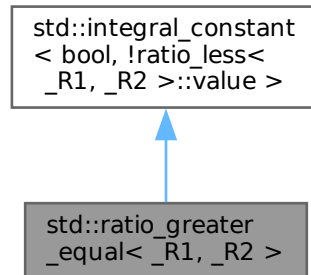
ratio_greater

The documentation for this struct was generated from the following file:

- [ratio](#)

6.866 std::ratio_greater_equal< _R1, _R2 > Struct Template Reference

Inheritance diagram for std::ratio_greater_equal< _R1, _R2 >:

**Public Types**

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.866.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_greater_equal< _R1, _R2 >
```

ratio_greater_equal

The documentation for this struct was generated from the following file:

- [ratio](#)

6.867 std::ratio_less< _R1, _R2 > Struct Template Reference

Inherits [__ratio_less_impl::type](#).

6.867.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_less< _R1, _R2 >
```

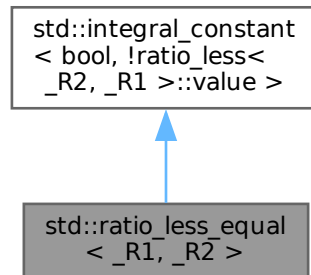
ratio_less

The documentation for this struct was generated from the following file:

- [ratio](#)

6.868 `std::ratio_less_equal<_R1, _R2>` Struct Template Reference

Inheritance diagram for `std::ratio_less_equal<_R1, _R2>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.868.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_less_equal<_R1, _R2>
```

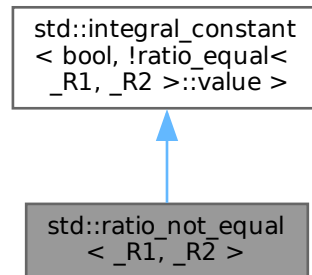
`ratio_less_equal`

The documentation for this struct was generated from the following file:

- [ratio](#)

6.869 `std::ratio_not_equal<_R1, _R2>` Struct Template Reference

Inheritance diagram for `std::ratio_not_equal<_R1, _R2>`:

**Public Types**

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr bool **value**

6.869.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_not_equal<_R1, _R2>
```

`ratio_not_equal`

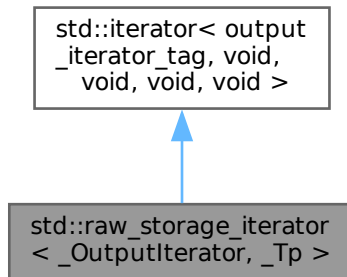
The documentation for this struct was generated from the following file:

- [ratio](#)

6.870 `std::raw_storage_iterator<_OutputIterator, _Tp>` Class Template Reference

```
#include <stl_raw_storage_iter.h>
```

Inheritance diagram for `std::raw_storage_iterator< _OutputIterator, _Tp >`:



Public Types

- typedef void [difference_type](#)
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

Public Member Functions

- [raw_storage_iterator](#) ([_OutputIterator](#) __x)
- [_OutputIterator](#) **base** () const
- [raw_storage_iterator](#) & **operator*** ()
- [raw_storage_iterator](#) & **operator++** ()
- [raw_storage_iterator](#) **operator++** (int)
- [raw_storage_iterator](#) & **operator=** ([_Tp](#) &&__element)
- [raw_storage_iterator](#) & **operator=** (const [_Tp](#) &__element)

Protected Attributes

- [_OutputIterator](#) **_M_iter**

6.870.1 Detailed Description

```
template<class _OutputIterator, class _Tp>
class std::raw_storage_iterator< _OutputIterator, _Tp >
```

This iterator class lets algorithms store their results into uninitialized memory.

6.870.2 Member Typedef Documentation

difference_type

```
typedef void std::iterator< output\_iterator\_tag , void , void , void , void >::difference_type
[inherited]
```

Distance between iterators is represented as this type.

iterator_category

```
typedef output\_iterator\_tag std::iterator< output\_iterator\_tag , void , void , void , void >←  
::iterator_category [inherited]
```

One of the [tag types](#).

pointer

```
typedef void std::iterator< output\_iterator\_tag , void , void , void , void >::pointer [inherited]
```

This type represents a pointer-to-value_type.

reference

```
typedef void std::iterator< output\_iterator\_tag , void , void , void , void >::reference [inherited]
```

This type represents a reference-to-value_type.

value_type

```
typedef void std::iterator< output\_iterator\_tag , void , void , void , void >::value_type [inherited]
```

The type "pointed to" by the iterator.

The documentation for this class was generated from the following file:

- [stl_raw_storage_iter.h](#)

6.871 `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >` Struct Template Reference

Inherits `std::_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >`.

Public Types

- `template<typename _Iter >`
 using **__same_value_type** = `is_same`< value_type, typename iterator_traits< _Iter >::value_type >
- `typedef std::_Rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >` **_Base**
- `template<typename _Compare2 >`
 using **Compatible_tree** = `_Rb_tree`< _Key, _Val, _KeyOfValue, _Compare2, _Alloc >
- `typedef _Base::allocator_type` **allocator_type**
- `typedef _Rb_tree_const_iterator< value_type >` **const_iterator**
- `typedef const value_type *` **const_pointer**
- `typedef const value_type &` **const_reference**
- `typedef std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- `typedef ptrdiff_t` **difference_type**
- using **insert_return_type** = `_Node_insert_return`< __conditional_t< `is_same_v`< _Key, _Val >, const_iterator, iterator >, [node_type](#) >
- `typedef _Rb_tree_iterator< value_type >` **iterator**
- `typedef _Key` **key_type**
- using **node_type** = `_Node_handle`< _Key, _Val, _Node_allocator >
- `typedef value_type *` **pointer**
- `typedef value_type &` **reference**
- `typedef std::reverse_iterator< iterator >` **reverse_iterator**
- `typedef size_t` **size_type**
- `typedef _Val` **value_type**

Public Member Functions

- **rb_tree** (const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- bool **__rb_verify** () const
- template<typename _Iterator >
void **_M_assign_equal** (_Iterator, _Iterator)
- template<typename _Iterator >
void **_M_assign_unique** (_Iterator, _Iterator)
- template<typename _Kt, typename _Req = __has_is_transparent_t<_Compare, _Kt>>
size_type **_M_count_tr** (const _Kt &__k) const
- template<typename... _Args>
iterator **_M_emplace_equal** (_Args &&... __args)
- template<typename... _Args>
auto **_M_emplace_equal** (_Args &&... __args) -> iterator
- template<typename... _Args>
iterator **_M_emplace_hint_equal** (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
auto **_M_emplace_hint_equal** (const_iterator __pos, _Args &&... __args) -> iterator
- template<typename... _Args>
iterator **_M_emplace_hint_unique** (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
auto **_M_emplace_hint_unique** (const_iterator __pos, _Args &&... __args) -> iterator
- template<typename... _Args>
[pair](#)< iterator, bool > **_M_emplace_unique** (_Args &&... __args)
- template<typename... _Args>
auto **_M_emplace_unique** (_Args &&... __args) -> [pair](#)< iterator, bool >
- template<typename _Kt, typename _Req = __has_is_transparent_t<_Compare, _Kt>>
[pair](#)< iterator, iterator > **_M_equal_range_tr** (const _Kt &__k)
- template<typename _Kt, typename _Req = __has_is_transparent_t<_Compare, _Kt>>
[pair](#)< const_iterator, const_iterator > **_M_equal_range_tr** (const _Kt &__k) const
- template<typename _Kt, typename _Req = __has_is_transparent_t<_Compare, _Kt>>
iterator **_M_find_tr** (const _Kt &__k)
- template<typename _Kt, typename _Req = __has_is_transparent_t<_Compare, _Kt>>
const_iterator **_M_find_tr** (const _Kt &__k) const
- [pair](#)< _Base_ptr, _Base_ptr > **_M_get_insert_equal_pos** (const key_type &__k)
- [pair](#)< _Base_ptr, _Base_ptr > **_M_get_insert_hint_equal_pos** (const_iterator __pos, const key_type &__k)
- [pair](#)< _Base_ptr, _Base_ptr > **_M_get_insert_hint_unique_pos** (const_iterator __pos, const key_type &__k)
- [pair](#)< _Base_ptr, _Base_ptr > **_M_get_insert_unique_pos** (const key_type &__k)
- const _Node_allocator & **_M_get_Node_allocator** () const noexcept
- _Node_allocator & **_M_get_Node_allocator** () noexcept
- template<typename _Arg >
iterator **_M_insert_equal** (_Arg &&__x)
- template<typename _Arg >
iterator **_M_insert_equal_** (const_iterator __pos, _Arg &&__x)
- template<typename _Arg, typename _NodeGen >
iterator **_M_insert_equal_** (const_iterator __pos, _Arg &&__x, _NodeGen &)
- template<typename _InputIterator >
__enable_if_t< __same_value_type< _InputIterator >::value > **_M_insert_range_equal** (_InputIterator __first, _InputIterator __last)
- template<typename _InputIterator >
__enable_if_t< !__same_value_type< _InputIterator >::value > **_M_insert_range_equal** (_InputIterator __first, _InputIterator __last)

- `template<typename _InputIterator >`
`__enable_if_t< __same_value_type< _InputIterator >::value > _M_insert_range_unique (_InputIterator __↵`
`first, _InputIterator __last)`
- `template<typename _InputIterator >`
`__enable_if_t<! __same_value_type< _InputIterator >::value > _M_insert_range_unique (_InputIterator __↵`
`first, _InputIterator __last)`
- `template<typename _Arg >`
`pair< typename _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >::iterator, bool > _M_insert_unique`
`(_Arg && __v)`
- `template<typename _Arg >`
`pair< iterator, bool > _M_insert_unique (_Arg && __x)`
- `template<typename _Arg >`
`iterator _M_insert_unique_ (const_iterator __pos, _Arg && __x)`
- `template<typename _Arg, typename _NodeGen >`
`iterator _M_insert_unique_ (const_iterator __pos, _Arg && __x, _NodeGen &)`
- `template<typename _Kt, typename _Req = __has_is_transparent_t< _Compare, _Kt>>`
`iterator _M_lower_bound_tr (const _Kt & __k)`
- `template<typename _Kt, typename _Req = __has_is_transparent_t< _Compare, _Kt>>`
`const_iterator _M_lower_bound_tr (const _Kt & __k) const`
- `template<typename _Compare2 >`
`void _M_merge_equal (_Compatible_tree< _Compare2 > & __src) noexcept`
- `template<typename _Compare2 >`
`void _M_merge_unique (_Compatible_tree< _Compare2 > & __src) noexcept`
- `iterator _M_reinsert_node_equal (node_type && __nh)`
- `iterator _M_reinsert_node_hint_equal (const_iterator __hint, node_type && __nh)`
- `iterator _M_reinsert_node_hint_unique (const_iterator __hint, node_type && __nh)`
- `insert_return_type _M_reinsert_node_unique (node_type && __nh)`
- `template<typename _Kt, typename _Req = __has_is_transparent_t< _Compare, _Kt>>`
`iterator _M_upper_bound_tr (const _Kt & __k)`
- `template<typename _Kt, typename _Req = __has_is_transparent_t< _Compare, _Kt>>`
`const_iterator _M_upper_bound_tr (const _Kt & __k) const`
- `const_iterator begin () const noexcept`
- `iterator begin () noexcept`
- `void clear () noexcept`
- `size_type count (const key_type & __k) const`
- `bool empty () const noexcept`
- `const_iterator end () const noexcept`
- `iterator end () noexcept`
- `pair< iterator, iterator > equal_range (const key_type & __k)`
- `pair< const_iterator, const_iterator > equal_range (const key_type & __k) const`
- `size_type erase (const key_type & __x)`
- `_GLIBCXX_ABI_TAG_CXX11 iterator erase (const_iterator __first, const_iterator __last)`
- `_GLIBCXX_ABI_TAG_CXX11 iterator erase (const_iterator __position)`
- `_GLIBCXX_ABI_TAG_CXX11 iterator erase (iterator __position)`
- `node_type extract (const key_type & __k)`
- `node_type extract (const_iterator __pos)`
- `iterator find (const key_type & __k)`
- `const_iterator find (const key_type & __k) const`
- `allocator_type get_allocator () const noexcept`
- `_Compare key_comp () const`
- `iterator lower_bound (const key_type & __k)`
- `const_iterator lower_bound (const key_type & __k) const`

- `size_type max_size ()` const noexcept
- `const_reverse_iterator rbegin ()` const noexcept
- `reverse_iterator rbegin ()` noexcept
- `const_reverse_iterator rend ()` const noexcept
- `reverse_iterator rend ()` noexcept
- `size_type size ()` const noexcept
- `void swap (_Rb_tree &__t)` noexcept(/*conditional */)
- `iterator upper_bound (const key_type &__k)`
- `const_iterator upper_bound (const key_type &__k)` const

Protected Types

- `typedef _Rb_tree_node_base * _Base_ptr`
- `typedef const _Rb_tree_node_base * _Const_Base_ptr`
- `typedef const _Rb_tree_node< _Val > * _Const_Link_type`
- `typedef _Rb_tree_node< _Val > * _Link_type`

Protected Member Functions

- `_Const_Link_type _M_begin ()` const noexcept
- `_Link_type _M_begin ()` noexcept
- `template<bool _MoveValue, typename _NodeGen >
_Link_type _M_clone_node (_Link_type __x, _NodeGen &__node_gen)`
- `template<typename... _Args>
void _M_construct_node (_Link_type __node, _Args &&... __args)`
- `template<typename... _Args>
_Link_type _M_create_node (_Args &&... __args)`
- `void _M_destroy_node (_Link_type __p)` noexcept
- `void _M_drop_node (_Link_type __p)` noexcept
- `_Const_Base_ptr _M_end ()` const noexcept
- `_Base_ptr _M_end ()` noexcept
- `_Link_type _M_get_node ()`
- `_Const_Base_ptr _M_leftmost ()` const noexcept
- `_Base_ptr & _M_leftmost ()` noexcept
- `_Link_type _M_mbegin ()` const noexcept
- `void _M_put_node (_Link_type __p)` noexcept
- `_Const_Base_ptr _M_rightmost ()` const noexcept
- `_Base_ptr & _M_rightmost ()` noexcept
- `_Const_Base_ptr _M_root ()` const noexcept
- `_Base_ptr & _M_root ()` noexcept

Static Protected Member Functions

- `static const _Key & _S_key (_Const_Base_ptr __x)`
- `static const _Key & _S_key (_Const_Link_type __x)`
- `static _Link_type _S_left (_Base_ptr __x)` noexcept
- `static _Const_Link_type _S_left (_Const_Base_ptr __x)` noexcept
- `static _Base_ptr _S_maximum (_Base_ptr __x)` noexcept
- `static _Const_Base_ptr _S_maximum (_Const_Base_ptr __x)` noexcept
- `static _Base_ptr _S_minimum (_Base_ptr __x)` noexcept
- `static _Const_Base_ptr _S_minimum (_Const_Base_ptr __x)` noexcept
- `static _Link_type _S_right (_Base_ptr __x)` noexcept
- `static _Const_Link_type _S_right (_Const_Base_ptr __x)` noexcept

Protected Attributes

- `_Rb_tree_impl<_Compare >_M_impl`

6.871.1 Detailed Description

```
template<class _Key, class _Value, class _KeyOfValue, class _Compare, class _Alloc = std::allocator<_↵
Value>>
struct __gnu_cxx::rb_tree<_Key, _Value, _KeyOfValue, _Compare, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

6.871.2 Member Function Documentation

`_M_merge_equal()`

```
template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
template<typename _Compare2 >
void std::_Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_merge_equal (
    _Compatible_tree<_Compare2 > & __src ) [inline], [noexcept], [inherited]
```

Merge from a compatible container into one with equivalent keys.

`_M_merge_unique()`

```
template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
template<typename _Compare2 >
void std::_Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_merge_unique (
    _Compatible_tree<_Compare2 > & __src ) [inline], [noexcept], [inherited]
```

Merge from a compatible container into one with unique keys.

`_M_reinsert_node_equal()`

```
template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
iterator std::_Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_reinsert_node_equal (
    node_type && __nh ) [inline], [inherited]
```

Re-insert an extracted node.

`_M_reinsert_node_hint_equal()`

```
template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
iterator std::_Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_reinsert_node_hint_equal
(
    const_iterator __hint,
    node_type && __nh ) [inline], [inherited]
```

Re-insert an extracted node.

`_M_reinsert_node_hint_unique()`

```
template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
```



```

iterator std::_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_reinsert_node_hint_unique
(
    const_iterator __hint,
    node_type && __nh ) [inline], [inherited]

```

Re-insert an extracted node.

`_M_reinsert_node_unique()`

```

template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
insert_return_type std::_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >::_M_reinsert_node↵
_unique (
    node_type && __nh ) [inline], [inherited]

```

Re-insert an extracted node.

`extract()` [1/2]

```

template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
node_type std::_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >::extract (
    const key_type & __k ) [inline], [inherited]

```

Extract a node.

`extract()` [2/2]

```

template<typename _Key , typename _Val , typename _KeyOfValue , typename _Compare , typename _↵
Alloc = allocator<_Val>>
node_type std::_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc >::extract (
    const_iterator __pos ) [inline], [inherited]

```

Extract a node.

The documentation for this struct was generated from the following file:

- [rb_tree](#)

6.872 `__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

```
#include <rb_tree_.hpp>
```

Inherits `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `base_type::const_pointer` **const_pointer**
- typedef `base_type::const_reference` **const_reference**
- typedef `base_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `rb_tree_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::key_const_pointer` **key_const_pointer**
- typedef `base_type::key_const_reference` **key_const_reference**

- `typedef base_type::key_pointer` **key_pointer**
- `typedef base_type::key_reference` **key_reference**
- `typedef base_type::key_type` **key_type**
- `typedef base_type::mapped_const_pointer` **mapped_const_pointer**
- `typedef base_type::mapped_const_reference` **mapped_const_reference**
- `typedef base_type::mapped_pointer` **mapped_pointer**
- `typedef base_type::mapped_reference` **mapped_reference**
- `typedef base_type::mapped_type` **mapped_type**
- `typedef __nothrowcopy::indicator` **no_throw_indicator**
- `typedef traits_type::node_const_iterator` **node_const_iterator**
- `typedef traits_type::node_iterator` **node_iterator**
- `typedef base_type::node_update` **node_update**
- `typedef base_type::const_iterator` **point_const_iterator**
- `typedef base_type::point_iterator` **point_iterator**
- `typedef base_type::pointer` **pointer**
- `typedef base_type::reference` **reference**
- `typedef base_type::reverse_iterator` **reverse_iterator**
- `typedef _Alloc::size_type` **size_type**
- `typedef integral_constant< int, Store_Hash >` **store_extra**
- `typedef stored_data< value_type, size_type, Store_Hash >` **stored_data_type**
- `typedef base_type::value_type` **value_type**

Public Member Functions

- **rb_tree_map** (const Cmp_Fn &)
- **rb_tree_map** (const Cmp_Fn &, const node_update &)
- **rb_tree_map** (const [direct_mask_range_hashing](#)< Size_Type > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- iterator **erase** (iterator)
- bool **erase** (key_const_reference)
- reverse_iterator **erase** (reverse_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- `std::pair< point_iterator, bool >` **insert** (const_reference)
- void **join** ([direct_mask_range_hashing](#)< Size_Type > &)
- point_iterator **lower_bound** (key_const_reference)
- point_const_iterator **lower_bound** (key_const_reference) const
- size_type **max_size** () const
- node_iterator [node_begin](#) ()
- node_const_iterator [node_begin](#) () const

- node_iterator [node_end](#) ()
- node_const_iterator [node_end](#) () const
- mapped_reference **operator[]** (key_const_reference r_key)
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- size_type **size** () const
- void **split** (key_const_reference, [direct_mask_range_hashing](#)< Size_Type > &)
- void **swap** ([direct_mask_range_hashing](#)< Size_Type > &)
- void **swap** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference)
- point_const_iterator **upper_bound** (key_const_reference) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Types

- typedef node_alloc_traits::value_type **node**
- typedef node_alloc_traits::allocator_type **node_allocator**
- typedef traits_type::null_node_update_pointer **null_node_update_pointer**
- typedef [types_traits](#)< Key, Mapped, _Alloc, false > **traits_base**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- template<typename Node_Update_>
void **apply_update** (node_pointer, Node_Update_*)
- void **apply_update** (node_pointer, null_node_update_pointer)
- [std::pair](#)< node_pointer, bool > **erase** (node_pointer)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, false_type)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, true_type)
- void **initialize_min_max** ()
- iterator **insert_imp_empty** (const_reference)
- [std::pair](#)< point_iterator, bool > **insert_leaf** (const_reference)
- iterator **insert_leaf_new** (const_reference, node_pointer, bool)
- void **join_finish** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- bool **join_prep** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- size_type **recursive_count** (node_pointer) const
- void **rotate_left** (node_pointer)
- void **rotate_parent** (node_pointer)
- void **rotate_right** (node_pointer)
- void **split_finish** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- bool **split_prep** (key_const_reference, [tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- void **update_min_max_for_erased_node** (node_pointer)
- template<typename Node_Update_>
void **update_to_top** (node_pointer, Node_Update_*)
- void **update_to_top** (node_pointer, null_node_update_pointer)
- void **value_swap** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)

Static Protected Member Functions

- static void **clear_imp** (node_pointer)

Protected Attributes

- node_pointer **m_p_head**
- size_type **m_size**

Static Protected Attributes

- static node_allocator **s_node_allocator**

6.872.1 Detailed Description

`template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _↵
_Alloc>`

`class __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`

Red-Black tree.

This implementation uses an idea from the SGI STL (using a *header* node which is needed for efficient iteration).

6.872.2 Member Function Documentation

node_begin() [1/2]

`template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >`

`node_iterator __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, ↵
_Alloc >::node_begin () [inline], [inherited]`

Returns a `node_iterator` corresponding to the node at the root of the tree.

node_begin() [2/2]

`template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >`

`node_const_iterator __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_↵
Traits, _Alloc >::node_begin () const [inline], [inherited]`

Returns a const `node_iterator` corresponding to the node at the root of the tree.

node_end() [1/2]

`template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >`

`node_iterator __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, ↵
_Alloc >::node_end () [inline], [inherited]`

Returns a `node_iterator` corresponding to a node just after a leaf of the tree.

node_end() [2/2]

`template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >`

`node_const_iterator __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_↵
Traits, _Alloc >::node_end () const [inline], [inherited]`

Returns a const `node_iterator` corresponding to a node just after a leaf of the tree.

The documentation for this class was generated from the following file:

- [rb_tree_.hpp](#)

6.873 `__gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc >` Struct Template Reference

```
#include <node.hpp>
```

Public Types

- typedef [rebind_traits](#)< _Alloc, metadata_type >::const_reference **metadata_const_reference**
- typedef [rebind_traits](#)< _Alloc, metadata_type >::reference **metadata_reference**
- typedef Metadata **metadata_type**
- typedef [rebind_traits](#)< _Alloc, [rb_tree_node_](#) >::pointer **node_pointer**
- typedef Value_Type **value_type**

Public Member Functions

- metadata_reference **get_metadata** ()
- metadata_const_reference **get_metadata** () const
- bool **special** () const

Public Attributes

- metadata_type **m_metadata**
- node_pointer **m_p_left**
- node_pointer **m_p_parent**
- node_pointer **m_p_right**
- bool **m_red**
- value_type **m_value**

6.873.1 Detailed Description

```
template<typename Value_Type, class Metadata, typename _Alloc>
struct __gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc >
```

Node for Red-Black trees.

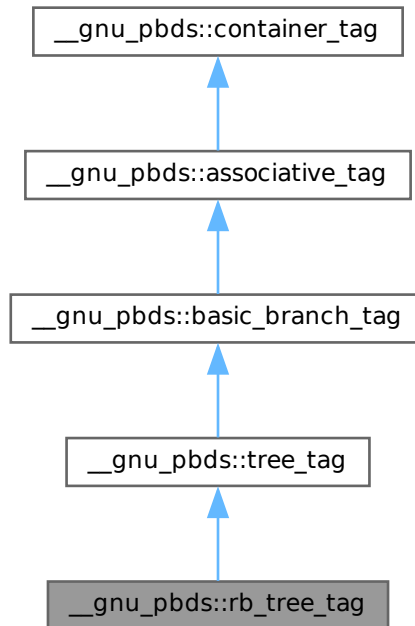
The documentation for this struct was generated from the following file:

- [rb_tree_map_/node.hpp](#)

6.874 `__gnu_pbds::rb_tree_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::rb_tree_tag:



6.874.1 Detailed Description

Red-black tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.875 __gnu_pbds::detail::rc< _Node, _Alloc > Class Template Reference

```
#include <rc.hpp>
```

Public Types

- typedef entry_const_pointer **const_iterator**
- typedef node_pointer **entry**

Public Member Functions

- **rc** (const [rc](#) &)
- const const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- const const_iterator **end** () const
- void **pop** ()

- void **push** (entry)
- size_type **size** () const
- void **swap** (rc &)
- node_pointer **top** () const

6.875.1 Detailed Description

```
template<typename _Node, typename _Alloc>
class __gnu_pbds::detail::rc< _Node, _Alloc >
```

Redundant binary counter.

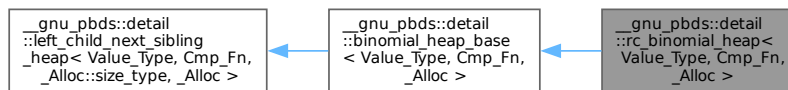
The documentation for this class was generated from the following file:

- [rc.hpp](#)

6.876 __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference

```
#include <rc_binomial_heap.hpp>
```

Inheritance diagram for __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >:



Public Types

- typedef base_type::allocator_type **allocator_type**
- typedef base_type::cmp_fn **cmp_fn**
- typedef [base_type::const_iterator](#) **const_iterator**
- typedef base_type::const_pointer **const_pointer**
- typedef base_type::const_reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef [base_type::iterator](#) **iterator**
- typedef [base_type::point_const_iterator](#) **point_const_iterator**
- typedef [base_type::point_iterator](#) **point_iterator**
- typedef base_type::pointer **pointer**
- typedef base_type::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef Value_Type **value_type**

Public Member Functions

- **rc_binomial_heap** (const [binomial_heap_base](#)< Value_Type, Cmp_Fn, _Alloc > &)
- **rc_binomial_heap** (const Cmp_Fn &)
- [iterator](#) **begin** ()
- [const_iterator](#) **begin** () const
- void **clear** ()
- bool **empty** () const

- `iterator end ()`
- `const_iterator end () const`
- `void erase (point_iterator)`
- `template<typename Pred >`
`size_type erase_if (Pred)`
- `Cmp_Fn & get_cmp_fn ()`
- `const Cmp_Fn & get_cmp_fn () const`
- `void join (binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > &)`
- `size_type max_size () const`
- `void modify (point_iterator, const_reference)`
- `void pop ()`
- `point_iterator push (const_reference)`
- `size_type size () const`
- `template<typename Pred >`
`void split (Pred, binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > &)`
- `void swap (binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > &)`
- `void swap (left_child_next_sibling_heap< Value_Type, Cmp_Fn, _Alloc::size_type, _Alloc > &)`
- `const_reference top () const`

Protected Types

- `typedef base_type::node node`
- `typedef alloc_traits::allocator_type node_allocator`
- `typedef _Alloc::size_type node_metadata`
- `typedef std::pair< node_pointer, node_pointer > node_pointer_pair`

Protected Member Functions

- `void actual_erase_node (node_pointer)`
- `void bubble_to_top (node_pointer)`
- `void clear_imp (node_pointer)`
- `template<typename It >`
`void copy_from_range (It, It)`
- `void find_max ()`
- `node_pointer get_new_node_for_insert (const_reference)`
- `node_pointer prune (Pred)`
- `void swap_with_parent (node_pointer, node_pointer)`
- `void to_linked_list ()`
- `void value_swap (left_child_next_sibling_heap &)`

Static Protected Member Functions

- `static void make_child_of (node_pointer, node_pointer)`
- `static node_pointer parent (node_pointer)`

Protected Attributes

- `node_pointer m_p_max`
- `node_pointer m_p_root`
- `size_type m_size`

6.876.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >
```

Redundant-counter binomial heap.

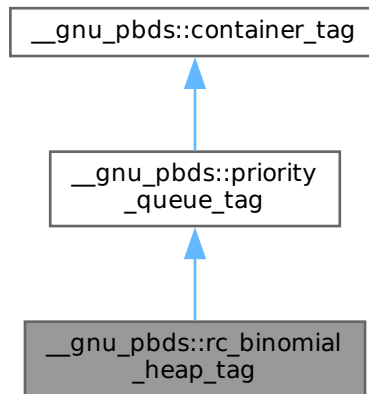
The documentation for this class was generated from the following file:

- [rc_binomial_heap.hpp](#)

6.877 __gnu_pbds::rc_binomial_heap_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::rc_binomial_heap_tag:



6.877.1 Detailed Description

Redundant-counter binomial-heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.878 __gnu_pbds::detail::rebind_traits< _Alloc, T > Struct Template Reference

```
#include <types_traits.hpp>
```

Inherits std::allocator_traits< _Alloc >::template rebind_traits.

Public Types

- using **const_reference** = const T &
- using **reference** = T &

6.878.1 Detailed Description

```
template<typename _Alloc, typename T>
struct __gnu_pbds::detail::rebind_traits< _Alloc, T >
```

Consistent API for accessing allocator-related types.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.879 std::filesystem::recursive_directory_iterator Class Reference

```
#include <fs_dir.h>
```

Public Types

- typedef ptrdiff_t **difference_type**
- typedef [input_iterator_tag](#) **iterator_category**
- typedef const [directory_entry](#) * **pointer**
- typedef const [directory_entry](#) & **reference**
- typedef [directory_entry](#) **value_type**

Public Member Functions

- **recursive_directory_iterator** (const [path](#) &__p)
- **recursive_directory_iterator** (const [path](#) &__p, [directory_options](#) __options)
- **recursive_directory_iterator** (const [path](#) &__p, [directory_options](#) __options, [error_code](#) &__ec)
- **recursive_directory_iterator** (const [path](#) &__p, [error_code](#) &__ec)
- **recursive_directory_iterator** (const [recursive_directory_iterator](#) &)=default
- **recursive_directory_iterator** ([recursive_directory_iterator](#) &&)=default
- int **depth** () const noexcept
- void **disable_recursion_pending** () noexcept
- [recursive_directory_iterator](#) & **increment** ([error_code](#) &__ec)
- const [directory_entry](#) & **operator*** () const noexcept
- [recursive_directory_iterator](#) & **operator++** ()
- [__directory_iterator_proxy](#) **operator++** (int)
- const [directory_entry](#) * **operator->** () const noexcept
- [recursive_directory_iterator](#) & **operator=** (const [recursive_directory_iterator](#) &__rhs) noexcept
- [recursive_directory_iterator](#) & **operator=** ([recursive_directory_iterator](#) &&__rhs) noexcept
- bool **operator==** ([default_sentinel_t](#)) const noexcept
- [directory_options](#) **options** () const noexcept
- void **pop** ()
- void **pop** ([error_code](#) &)
- bool **recursion_pending** () const noexcept

Friends

- uintmax_t **filesystem::remove_all** (const [path](#) &)
- uintmax_t **filesystem::remove_all** (const [path](#) &, [error_code](#) &)
- bool **operator==** (const [recursive_directory_iterator](#) &__lhs, const [recursive_directory_iterator](#) &__rhs) noexcept

Related Symbols

(Note that these are not member symbols.)

- [recursive_directory_iterator begin](#) ([recursive_directory_iterator](#) __iter) noexcept
- [recursive_directory_iterator end](#) ([recursive_directory_iterator](#)) noexcept

6.879.1 Detailed Description

Iterator type for recursively traversing a directory hierarchy.

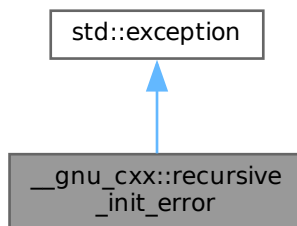
The documentation for this class was generated from the following file:

- [bits/fs_dir.h](#)

6.880 __gnu_cxx::recursive_init_error Class Reference

```
#include <cxxabi.h>
```

Inheritance diagram for __gnu_cxx::recursive_init_error:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.880.1 Detailed Description

Exception thrown by `__cxa_guard_acquire`.

C++ 2011 6.7 [stmt.dcl]/4: If control re-enters the declaration recursively while the variable is being initialized, the behavior is undefined.

Since we already have a library function to handle locking, we might as well check for this situation and throw an exception. We use the second byte of the guard variable to remember that we're in the middle of an initialization.

6.880.2 Member Function Documentation

what()

```
virtual const char * std::exception::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error.

Reimplemented in [std::bad_alloc](#), [std::ios_base::failure](#), [std::bad_exception](#), [std::bad_cast](#), [std::bad_typeid](#), [std::bad_any_cast](#), [std::filesystem::filesystem_error](#), [std::bad_weak_ptr](#), [std::bad_function_call](#), [std::experimental::fundamentals_v1::bad_weak_ptr](#), [std::experimental::filesystem_v1::filesystem_error](#), [std::future_error](#), [std::logic_error](#), [std::runtime_error](#), and [std::bad_optional_access](#).

The documentation for this class was generated from the following file:

- [cxxabi.h](#)

6.881 std::recursive_mutex Class Reference

Inherits `std::__recursive_mutex_base`.

Public Types

- typedef `__native_type * native_handle_type`

Public Member Functions

- `recursive_mutex` (const [recursive_mutex](#) &)=delete
- void `lock` ()
- `native_handle_type native_handle` () noexcept
- [recursive_mutex](#) & `operator=` (const [recursive_mutex](#) &)=delete
- bool `try_lock` () noexcept
- void `unlock` ()

6.881.1 Detailed Description

The standard recursive mutex type.

The documentation for this class was generated from the following file:

- [mutex](#)

6.882 std::recursive_timed_mutex Class Reference

Public Member Functions

- `recursive_timed_mutex` (const [recursive_timed_mutex](#) &)=delete
- void `lock` ()
- [recursive_timed_mutex](#) & `operator=` (const [recursive_timed_mutex](#) &)=delete
- bool `try_lock` ()
- template<typename `_Rep`, typename `_Period` >
bool `try_lock_for` (const [chrono::duration](#)< `_Rep`, `_Period` > &__rtime)
- template<typename `_Clock`, typename `_Duration` >
bool `try_lock_until` (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__atime)
- void `unlock` ()

6.882.1 Detailed Description

`recursive_timed_mutex`

The documentation for this class was generated from the following file:

- [mutex](#)

6.883 std::bitset<_Nb>::reference Class Reference

Public Member Functions

- `reference` ([bitset](#) &__b, size_t __pos) noexcept
- `reference` (const [reference](#) &)=default
- [reference](#) & `flip` () noexcept

- **operator bool** () const noexcept
- [reference](#) & **operator=** (bool __x) noexcept
- [reference](#) & **operator=** (const [reference](#) &__j) noexcept
- bool **operator~** () const noexcept

Friends

- class **bitset**

6.883.1 Detailed Description

```
template<size_t _Nb>
class std::bitset<_Nb>::reference
```

This encapsulates the concept of a single bit. An instance of this class is a proxy for an actual bit; this way the individual bit operations are done as faster word-size bitwise instructions.

Most users will never need to use this class directly; conversions to and from bool are automatic and should be transparent. Overloaded operators help to preserve the illusion.

(On a typical system, this *bit reference* is 64 times the size of an actual bit. Ha.)

The documentation for this class was generated from the following file:

- [bitset](#)

6.884 std::tr2::dynamic_bitset<_WordT, _Alloc>::reference Class Reference

Public Member Functions

- **reference** ([dynamic_bitset](#) &__b, size_type __pos) noexcept
- [reference](#) & **flip** () noexcept
- **operator bool** () const noexcept
- [reference](#) & **operator=** (bool __x) noexcept
- [reference](#) & **operator=** (const [reference](#) &__j) noexcept
- bool **operator~** () const noexcept

Friends

- class **dynamic_bitset**

6.884.1 Detailed Description

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
class std::tr2::dynamic_bitset<_WordT, _Alloc>::reference
```

This encapsulates the concept of a single bit. An instance of this class is a proxy for an actual bit; this way the individual bit operations are done as faster word-size bitwise instructions.

Most users will never need to use this class directly; conversions to and from bool are automatic and should be transparent. Overloaded operators help to preserve the illusion.

(On a typical system, this "bit %reference" is 64 times the size of an actual bit. Ha.)

The documentation for this class was generated from the following file:

- [dynamic_bitset](#)

6.885 std::reference_wrapper<_Tp> Class Template Reference

```
#include <refwrap.h>
```

Public Types

- typedef `_Tp` `type`

Public Member Functions

- template<typename `_Up` , typename = `__not_same<_Up>`, typename = `decltype(reference_wrapper::_S_fun(std::declval<_Up>()))`> constexpr `reference_wrapper` (`_Up` && `__uref`) noexcept(noexcept(`reference_wrapper::_S_fun(std::declval<_Up>())`))
- `reference_wrapper` (const `reference_wrapper` &)=default
- constexpr `_Tp` & `get` () const noexcept
- constexpr `operator _Tp &` () const noexcept
- template<typename... `_Args`> constexpr `__invoke_result< _Tp &, _Args... >::type operator()` (`_Args` &&... `__args`) const noexcept(`__is_nothrow_invocable< _Tp &, _Args... >::value`)
- `reference_wrapper` & `operator=` (const `reference_wrapper` &)=default

Related Symbols

(Note that these are not member symbols.)

- template<typename `_Tp` > constexpr `reference_wrapper<_Tp>` `ref` (`_Tp` & `__t`) noexcept
- template<typename `_Tp` > constexpr `reference_wrapper< const _Tp >` `cref` (const `_Tp` & `__t`) noexcept
- template<typename `_Tp` > constexpr `reference_wrapper<_Tp>` `ref` (`reference_wrapper<_Tp>` & `__t`) noexcept
- template<typename `_Tp` > constexpr `reference_wrapper< const _Tp >` `cref` (`reference_wrapper<_Tp>` & `__t`) noexcept

6.885.1 Detailed Description

```
template<typename _Tp>
class std::reference_wrapper<_Tp>
```

Primary class template for `reference_wrapper`.

6.885.2 Friends And Related Symbol Documentation

`cref()` [1/2]

```
template<typename _Tp >
constexpr reference_wrapper< const _Tp > cref (
    const _Tp & __t ) [related]
```

Denotes a const reference should be taken to a variable.

`cref()` [2/2]

```
template<typename _Tp >
constexpr reference_wrapper< const _Tp > cref (
    reference_wrapper<_Tp> & __t ) [related]
```

`std::cref` overload to prevent wrapping a `reference_wrapper`

ref() [1/2]

```
template<typename _Tp >
constexpr reference\_wrapper< _Tp > ref (
    _Tp & __t ) [related]
```

Denotes a reference should be taken to a variable.

ref() [2/2]

```
template<typename _Tp >
constexpr reference\_wrapper< _Tp > ref (
    reference\_wrapper< _Tp > __t ) [related]
```

std::ref overload to prevent wrapping a reference_wrapper

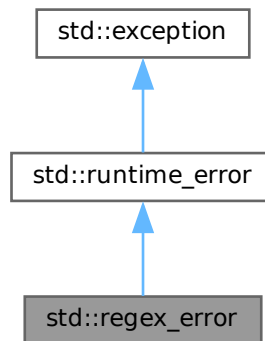
The documentation for this class was generated from the following files:

- [type_traits](#)
- [refwrap.h](#)

6.886 std::regex_error Class Reference

```
#include <regex_error.h>
```

Inheritance diagram for std::regex_error:

**Public Member Functions**

- [regex_error](#) ([error_type](#) __ecode)
- [regex_constants::error_type](#) code () const noexcept
- virtual const char * [what](#) () const noexcept

Friends

- void [__throw_regex_error](#) ([error_type](#) __ecode, const char * __what)

6.886.1 Detailed Description

A regular expression exception class.

The regular expression library throws objects of this class on error.

6.886.2 Constructor & Destructor Documentation

`regex_error()`

```
std::regex_error::regex_error (
    error_type __ecode ) [explicit]
```

Constructs a `regex_error` object.

Parameters

<code>__ecode</code>	the <code>regex_error</code> code.
----------------------	------------------------------------

6.886.3 Member Function Documentation

`code()`

```
regex_constants::error_type std::regex_error::code ( ) const [inline], [noexcept]
```

Gets the `regex_error` code.

Returns

the `regex_error` code.

`what()`

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

The documentation for this class was generated from the following file:

- [regex_error.h](#)

6.887 `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >` Class Template Reference

```
#include <regex.h>
```

Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef `const value_type *` **pointer**
- typedef `const value_type &` **reference**
- typedef [basic_regex< _Ch_type, _Rx_traits >](#) **regex_type**
- typedef [match_results< _Bi_iter >](#) **value_type**

Public Member Functions

- [regex_iterator](#) ()=default
- [regex_iterator](#) ([_Bi_iter](#) __a, [_Bi_iter](#) __b, const [regex_type](#) &__re, [regex_constants::match_flag_type](#) __m↵
m=[regex_constants::match_default](#))
- **regex_iterator** ([_Bi_iter](#), [_Bi_iter](#), const [regex_type](#) &&, [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))=delete
- [regex_iterator](#) (const [regex_iterator](#) &)=default
- `const value_type & operator*` () const noexcept

- [regex_iterator](#) & [operator++](#) ()
- [regex_iterator](#) [operator++](#) (int)
- const value_type * [operator->](#) () const noexcept
- [regex_iterator](#) & [operator=](#) (const [regex_iterator](#) &)=default
- bool [operator==](#) (const [regex_iterator](#) &) const noexcept
- bool [operator==](#) ([default_sentinel_t](#)) const noexcept

6.887.1 Detailed Description

```
template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename
_Rx_traits = regex_traits<_Ch_type>>
class std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >
```

An iterator adaptor that will provide repeated calls of `regex_search` over a range until no more matches remain.

6.887.2 Constructor & Destructor Documentation

[regex_iterator\(\)](#) [1/3]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::regex_iterator ( ) [default]
```

Provides a singular iterator, useful for indicating one-past-the-end of a range.

Referenced by [std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::regex_iterator\(\)](#).

[regex_iterator\(\)](#) [2/3]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::regex_iterator (
    _Bi_iter __a,
    _Bi_iter __b,
    const regex_type & __re,
    regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a `regex_iterator`...

Parameters

↩ <code>__a</code>	[IN] The start of a text range to search.
↩ <code>__b</code>	[IN] One-past-the-end of the text range to search.
↩ <code>__re</code>	[IN] The regular expression to match.
↩ <code>__m</code>	[IN] Policy flags for match rules.

References [std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::regex_iterator\(\)](#), and [std::regex_search\(\)](#).

[regex_iterator\(\)](#) [3/3]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::regex_iterator (
    const regex_iterator<_Bi_iter, _Ch_type, _Rx_traits > & ) [default]
```

Copy constructs a regex_iterator.

6.887.3 Member Function Documentation

operator*()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
const value_type & std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator* ( ) const
[inline], [noexcept]
```

Dereferences a regex_iterator.

operator++() [1/2]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_iterator & std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++ ( )
```

Increments a regex_iterator.

operator++() [2/2]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_iterator std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++ (
    int ) [inline]
```

Postincrements a regex_iterator.

operator->()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
const value_type * std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator-> ( ) const
[inline], [noexcept]
```

Selects a regex_iterator member.

operator=()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_iterator & std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator= (
    const regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & ) [default]
```

Copy assigns one regex_iterator to another.

operator==()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
bool std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator== (
    const regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & ) const [noexcept]
```

Tests the equivalence of two regex iterators.

The documentation for this class was generated from the following file:

- [regex.h](#)

6.888 std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference

```
#include <regex.h>
```

Public Types

- typedef std::ptrdiff_t **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef const [value_type](#) * **pointer**
- typedef const [value_type](#) & **reference**
- typedef [basic_regex](#)< _Ch_type, _Rx_traits > **regex_type**
- typedef [sub_match](#)< _Bi_iter > **value_type**

Public Member Functions

- [regex_token_iterator](#) ()
- template<std::size_t _Nm>
[regex_token_iterator](#) (_Bi_iter __a, _Bi_iter __b, const [regex_type](#) &__re, const int(&__submatches)[_Nm], [regex_constants::match_flag_type](#) __m=[regex_constants::match_default](#))
- [regex_token_iterator](#) (_Bi_iter __a, _Bi_iter __b, const [regex_type](#) &__re, const [std::vector](#)< int > &__submatches, [regex_constants::match_flag_type](#) __m=[regex_constants::match_default](#))
- [regex_token_iterator](#) (_Bi_iter __a, _Bi_iter __b, const [regex_type](#) &__re, [initializer_list](#)< int > __submatches, [regex_constants::match_flag_type](#) __m=[regex_constants::match_default](#))
- [regex_token_iterator](#) (_Bi_iter __a, _Bi_iter __b, const [regex_type](#) &__re, int __submatch=0, [regex_constants::match_flag_type](#) __m=[regex_constants::match_default](#))
- template<std::size_t _Nm>
regex_token_iterator (_Bi_iter, _Bi_iter, const [regex_type](#) &&, const int(&)[_Nm], [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))
- **regex_token_iterator** (_Bi_iter, _Bi_iter, const [regex_type](#) &&, const [std::vector](#)< int > &, [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))
- **regex_token_iterator** (_Bi_iter, _Bi_iter, const [regex_type](#) &&, [initializer_list](#)< int >, [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))
- **regex_token_iterator** (_Bi_iter, _Bi_iter, const [regex_type](#) &&, int=0, [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))
- [regex_token_iterator](#) (const [regex_token_iterator](#) &__rhs)
- const [value_type](#) & **operator*** () const
- [regex_token_iterator](#) & **operator++** ()
- [regex_token_iterator](#) **operator++** (int)
- const [value_type](#) * **operator->** () const
- [regex_token_iterator](#) & **operator=** (const [regex_token_iterator](#) &__rhs)
- bool **operator==** (const [regex_token_iterator](#) &__rhs) const
- bool **operator==** ([default_sentinel_t](#)) const noexcept

6.888.1 Detailed Description

```
template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename
_Rx_traits = regex_traits<_Ch_type>>
class std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >
```

Iterates over submatches in a range (or *splits* a text string).

The purpose of this iterator is to enumerate all, or all specified, matches of a regular expression within a text range. The dereferenced value of an iterator of this class is a `std::sub_match` object.

6.888.2 Constructor & Destructor Documentation

regex_token_iterator() [1/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>>
```

```
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator ( ) [inline]
```

Default constructs a regex_token_iterator.

A default-constructed regex_token_iterator is a singular iterator that will compare equal to the one-past-the-end value for any iterator of the same type.

regex_token_iterator() [2/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>>
```

```
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator (
    _Bi_iter __a,
    _Bi_iter __b,
    const regex_type & __re,
    int __submatch = 0,
    regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__submatch</code>	[IN] Which submatch to return. There are some special values for this parameter: <ul style="list-style-type: none"> • -1 each enumerated subexpression does NOT match the regular expression (aka field splitting) • 0 the entire string matching the subexpression is returned for each match within the text. • >0 enumerates only the indicated subexpression from a match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

regex_token_iterator() [3/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>>
```

```
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator (
    _Bi_iter __a,
    _Bi_iter __b,
    const regex_type & __re,
    const std::vector< int > & __submatches,
    regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.

Parameters

<code>__re</code>	[IN] The regular expression to search for.
<code>__submatches</code>	[IN] A list of subexpressions to return for each regular expression match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

regex_token_iterator() [4/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator (
    _Bi_iter __a,
    _Bi_iter __b,
    const regex_type & __re,
    initializer_list< int > __submatches,
    regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__submatches</code>	[IN] A list of subexpressions to return for each regular expression match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

regex_token_iterator() [5/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
template<std::size_t _Nm>
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator (
    _Bi_iter __a,
    _Bi_iter __b,
    const regex_type & __re,
    const int (&) __submatches[_Nm],
    regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__submatches</code>	[IN] A list of subexpressions to return for each regular expression match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

regex_token_iterator() [6/6]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator (
    const regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs ) [inline]
```

Copy constructs a regex_token_iterator.

Parameters

<code>__rhs</code>	[IN] A regex_token_iterator to copy.
--------------------	--------------------------------------

6.888.3 Member Function Documentation**operator*()**

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
const value_type & std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator* ( )
const [inline]
```

Dereferences a regex_token_iterator.

operator++() [1/2]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_token_iterator & std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++ (
    )
```

Increments a regex_token_iterator.

operator++() [2/2]

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_token_iterator std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++ (
    int ) [inline]
```

Postincrements a regex_token_iterator.

operator->()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
const value_type * std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator-> ( )
const [inline]
```

Selects a regex_token_iterator member.

operator=()

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
regex_token_iterator & std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator= (
    const regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs )
```

Assigns a regex_token_iterator to another.

Parameters

<code>__rhs</code>	[IN] A <code>regex_token_iterator</code> to copy.
--------------------	---

`operator==()`

```
template<typename _Bi_iter , typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type,
typename _Rx_traits = regex_traits<_Ch_type>>
bool std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator== (
    const regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs ) const
```

Compares a `regex_token_iterator` to another for equality.

The documentation for this class was generated from the following file:

- [regex.h](#)

6.889 `std::regex_traits< _Ch_type >` Class Template Reference

```
#include <regex.h>
```

Public Types

- typedef `_RegexMask` **`char_class_type`**
- typedef `_Ch_type` **`char_type`**
- typedef [std::locale](#) **`locale_type`**
- typedef [std::basic_string](#)< `char_type` > **`string_type`**

Public Member Functions

- [regex_traits](#) ()
- [locale_type](#) `getloc` () const
- [locale_type](#) `imbue` ([locale_type](#) __loc)
- bool [isctype](#) (`_Ch_type` __c, `char_class_type` __f) const
- template<typename `_Fwd_iter` >
`char_class_type` [lookup_classname](#) (`_Fwd_iter` __first, `_Fwd_iter` __last, bool __icase=false) const
- template<typename `_Fwd_iter` >
[string_type](#) [lookup_collatename](#) (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- template<typename `_Fwd_iter` >
[string_type](#) [transform](#) (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- template<typename `_Fwd_iter` >
[string_type](#) [transform_primary](#) (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- `char_type` [translate](#) (`char_type` __c) const
- `char_type` [translate_nocase](#) (`char_type` __c) const
- int [value](#) (`_Ch_type` __ch, int __radix) const

Static Public Member Functions

- static `std::size_t` [length](#) (const `char_type` *__p)

Protected Attributes

- [locale_type](#) `_M_locale`

6.889.1 Detailed Description

template<typename _Ch_type>
class `std::regex_traits<_Ch_type>`

Describes aspects of a regular expression.

A regular expression traits class that satisfies the requirements of section [28.7].

The class `regex` is parameterized around a set of related types and functions used to complete the definition of its semantics. This class satisfies the requirements of such a traits class.

6.889.2 Constructor & Destructor Documentation

`regex_traits()`

```
template<typename _Ch_type >
std::regex_traits<_Ch_type>::regex_traits ( ) [inline]
```

Constructs a default traits object.

6.889.3 Member Function Documentation

`getloc()`

```
template<typename _Ch_type >
locale_type std::regex_traits<_Ch_type>::getloc ( ) const [inline]
```

Gets a copy of the current locale in use by the `regex_traits` object.

`imbue()`

```
template<typename _Ch_type >
locale_type std::regex_traits<_Ch_type>::imbue (
    locale_type __loc ) [inline]
```

Imbues the `regex_traits` object with a copy of a new locale.

Parameters

<code>__loc</code>	A locale.
--------------------	-----------

Returns

a copy of the previous locale in use by the `regex_traits` object.

Note

Calling `imbue` with a different locale than the one currently in use invalidates all cached data held by `*this`.

References [std::swap\(\)](#).

`isctype()`

```
template<typename _Ch_type >
bool std::regex_traits<_Ch_type>::isctype (
    _Ch_type __c,
    char_class_type __f ) const
```

Determines if `c` is a member of an identified class.

Parameters

<code>__c</code>	a character.
<code>__f</code>	a class type (as returned from <code>lookup_classname</code>).

Returns

true if the character `__c` is a member of the classification represented by `__f`, false otherwise.

Exceptions

<code>std::bad_cast</code>	if the current locale does not have a ctype facet.
----------------------------	--

length()

```
template<typename _Ch_type >
static std::size_t std::regex_traits< _Ch_type >::length (
    const char_type * __p ) [inline], [static]
```

Gives the length of a C-style string starting at `__p`.

Parameters

<code>__p</code>	a pointer to the start of a character sequence.
------------------	---

Returns

the number of characters between `*__p` and the first default-initialized value of type `char_type`. In other words, uses the C-string algorithm for determining the length of a sequence of characters.

References [std::basic_string< _CharT, _Traits, _Alloc >::length\(\)](#).

lookup_classname()

```
template<typename _Ch_type >
template<typename _Fwd_iter >
char_class_type std::regex_traits< _Ch_type >::lookup_classname (
    _Fwd_iter __first,
    _Fwd_iter __last,
    bool __icase = false ) const
```

Maps one or more characters to a named character classification.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.
<code>__icase</code>	ignores the case of the classification name.

Returns

an unspecified value that represents the character classification named by the character sequence designated by the iterator range `[__first, __last)`. If `icase` is true, the returned mask identifies the classification regardless of the case of the characters to be matched (for example, `[:lower:]` is the same as `[:alpha:]`), otherwise a case-dependent classification is returned. The value returned shall be independent of the case of the characters in the character sequence. If the name is not recognized then returns a value that compares equal to 0.

At least the following names (or their wide-character equivalent) are supported.

- d
- w
- s
- alnum
- alpha
- blank
- cntrl
- digit
- graph
- lower
- print
- punct
- space
- upper
- xdigit

lookup_collatename()

```
template<typename _Ch_type >
template<typename _Fwd_iter >
string_type std::regex_traits< _Ch_type >::lookup_collatename (
    _Fwd_iter __first,
    _Fwd_iter __last ) const
```

Gets a collation element by name.

Parameters

<code>__first</code>	beginning of the collation element name.
<code>__last</code>	one-past-the-end of the collation element name.

Returns

a sequence of one or more characters that represents the collating element consisting of the character sequence designated by the iterator range `[__first, __last)`. Returns an empty string if the character sequence is not a valid collating element.

transform()

```
template<typename _Ch_type >
template<typename _Fwd_iter >
string_type std::regex_traits< _Ch_type >::transform (
    _Fwd_iter __first,
    _Fwd_iter __last ) const [inline]
```

Gets a sort key for a character sequence.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.

Returns a sort key for the character sequence designated by the iterator range [F1, F2) such that if the character sequence [G1, G2) sorts before the character sequence [H1, H2) then `v.transform(G1, G2) < v.transform(H1, H2)`. What this really does is provide a more efficient way to compare a string to multiple other strings in locales with fancy collation rules and equivalence classes.

Returns

a locale-specific sort key equivalent to the input range.

Exceptions

<code>std::bad_cast</code>	if the current locale does not have a collate facet.
----------------------------	--

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.
Referenced by `std::regex_traits< _Ch_type >::transform_primary()`.

transform_primary()

```
template<typename _Ch_type >
template<typename _Fwd_iter >
string_type std::regex_traits< _Ch_type >::transform_primary (
    _Fwd_iter __first,
    _Fwd_iter __last ) const [inline]
```

Gets a sort key for a character sequence, independent of case.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.

Effects: if `typeid(use_facet<collate<_Ch_type>>) == typeid(collate_byname<_Ch_type>)` and the form of the sort key returned by `collate_byname<_Ch_type>::transform(__first, __last)` is known and can be converted into a primary sort key then returns that key, otherwise returns an empty string.

Todo Implement this function correctly.

References `std::regex_traits< _Ch_type >::transform()`.

translate()

```
template<typename _Ch_type >
```

```
char_type std::regex_traits<_Ch_type>::translate (
    char_type __c ) const [inline]
```

Performs the identity translation.

Parameters

<code>__c</code>	A character to the locale-specific character set.
------------------	---

Returns

`__c`.

translate_nocase()

```
template<typename _Ch_type>
char_type std::regex_traits<_Ch_type>::translate_nocase (
    char_type __c ) const [inline]
```

Translates a character into a case-insensitive equivalent.

Parameters

<code>__c</code>	A character to the locale-specific character set.
------------------	---

Returns

the locale-specific lower-case equivalent of `__c`.

Exceptions

<code>std::bad_cast</code>	if the imbued locale does not support the ctype facet.
----------------------------	--

value()

```
template<typename _Ch_type>
int std::regex_traits<_Ch_type>::value (
    _Ch_type __ch,
    int __radix ) const
```

Converts a digit to an int.

Parameters

<code>__ch</code>	a character representing a digit.
<code>__radix</code>	the radix if the numeric conversion (limited to 8, 10, or 16).

Returns

the value represented by the digit `__ch` in base `radix` if the character `__ch` is a valid digit in base `radix`; otherwise returns -1.

The documentation for this class was generated from the following file:

- [regex.h](#)

6.890 `std::remove_all_extents< _Tp >` Struct Template Reference**Public Types**

- `typedef _Tp type`

6.890.1 Detailed Description

```
template<typename _Tp>
struct std::remove_all_extents< _Tp >
```

`remove_all_extents`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.891 `std::remove_const< _Tp >` Struct Template Reference**Public Types**

- `typedef _Tp type`

6.891.1 Detailed Description

```
template<typename _Tp>
struct std::remove_const< _Tp >
```

`remove_const`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.892 `std::remove_cv< _Tp >` Struct Template Reference

Inherited by `std::remove_cvref< _Tp >`, `std::remove_cvref< _Tp & >`, and `std::remove_cvref< _Tp && >`.

Public Types

- `using type = _Tp`

6.892.1 Detailed Description

```
template<typename _Tp>
struct std::remove_cv< _Tp >
```

`remove_cv`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.893 `std::remove_extent< _Tp >` Struct Template Reference

Public Types

- `typedef _Tp type`

6.893.1 Detailed Description

```
template<typename _Tp>  
struct std::remove_extent< _Tp >
```

`remove_extent`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.894 `std::remove_pointer< _Tp >` Struct Template Reference

Inherits `std::__remove_pointer_helper< _Tp, typename >`.

Public Types

- `typedef _Tp type`

6.894.1 Detailed Description

```
template<typename _Tp>  
struct std::remove_pointer< _Tp >
```

`remove_pointer`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.895 `std::remove_reference< _Tp >` Struct Template Reference

Public Types

- `typedef _Tp type`

6.895.1 Detailed Description

```
template<typename _Tp>  
struct std::remove_reference< _Tp >
```

`remove_reference`

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.896 `std::remove_volatile< _Tp >` Struct Template Reference

Public Types

- `typedef _Tp type`

6.896.1 Detailed Description

```
template<typename _Tp>
struct std::remove_volatile<_Tp >
```

remove_volatile

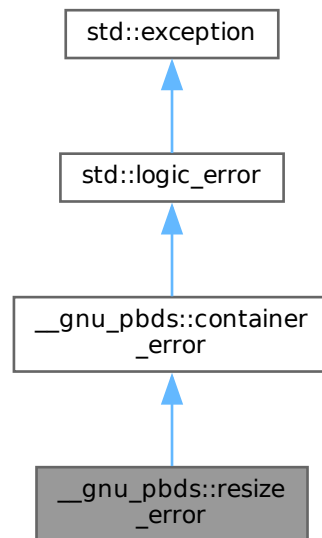
The documentation for this struct was generated from the following file:

- [type_traits](#)

6.897 __gnu_pbds::resize_error Struct Reference

```
#include <exception.hpp>
```

Inheritance diagram for __gnu_pbds::resize_error:



Public Member Functions

- virtual const char * [what](#) () const noexcept

6.897.1 Detailed Description

A container cannot be resized.

6.897.2 Member Function Documentation

what()

```
virtual const char * std::logic_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

6.898 `__gnu_pbds::detail::resize_policy<_Tp>` Class Template Reference

```
#include <resize_policy.hpp>
```

Public Types

- typedef `_Tp` `size_type`

Public Member Functions

- `resize_policy` (const [resize_policy](#) &other)
- `size_type` `get_new_size_for_arbitrary` (`size_type`) const
- `size_type` `get_new_size_for_grow` () const
- `size_type` `get_new_size_for_shrink` () const
- bool `grow_needed` (`size_type`) const
- void `notify_arbitrary` (`size_type`)
- void `notify_grow_resize` ()
- void `notify_shrink_resize` ()
- bool `resize_needed_for_grow` (`size_type`) const
- bool `resize_needed_for_shrink` (`size_type`) const
- bool `shrink_needed` (`size_type`) const
- void `swap` ([resize_policy](#)<`_Tp`> &)

Static Public Attributes

- static const `_Tp` `min_size`

6.898.1 Detailed Description

```
template<typename _Tp>  
class __gnu_pbds::detail::resize_policy<_Tp>
```

Resize policy for binary heap.

The documentation for this class was generated from the following file:

- [resize_policy.hpp](#)

6.899 `std::result_of<_Signature>` Struct Template Reference

6.899.1 Detailed Description

```
template<typename _Signature>  
struct std::result_of<_Signature>
```

`result_of`

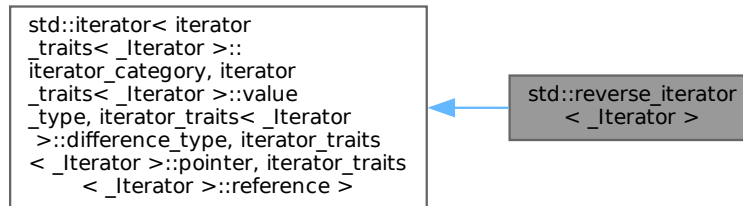
The documentation for this struct was generated from the following file:

- [type_traits](#)

6.900 std::reverse_iterator< _Iterator > Class Template Reference

```
#include <stl_iterator.h>
```

Inheritance diagram for std::reverse_iterator< _Iterator >:



Public Types

- using **difference_type** = iter_difference_t< _Iterator >
- using **iterator_category** = __detail::__clamp_iter_cat< typename __traits_type::iterator_category, [random_access_iterator_tag](#) >
- using **iterator_concept** = __conditional_t< random_access_iterator< _Iterator >, [random_access_iterator_tag](#), [bidirectional_iterator_tag](#) >
- typedef _Iterator **iterator_type**
- typedef __traits_type::pointer **pointer**
- using **reference** = iter_reference_t< _Iterator >
- using **value_type** = iter_value_t< _Iterator >

Public Member Functions

- constexpr [reverse_iterator](#) () noexcept(*/*conditional */*)
- constexpr [reverse_iterator](#) (const [reverse_iterator](#) &__x) noexcept(*/*conditional */*)
- template<typename _Iter >
requires __convertible<_Iter>
constexpr [reverse_iterator](#) (const [reverse_iterator](#)< _Iter > &__x) noexcept(*/*conditional */*)
- constexpr [reverse_iterator](#) (iterator_type __x) noexcept(*/*conditional */*)
- constexpr iterator_type [base](#) () const noexcept(*/*conditional */*)
- constexpr reference [operator*](#) () const
- constexpr [reverse_iterator](#) [operator+](#) (difference_type __n) const
- constexpr [reverse_iterator](#) & [operator++](#) ()
- constexpr [reverse_iterator](#) [operator++](#) (int)
- constexpr [reverse_iterator](#) & [operator+=](#) (difference_type __n)
- constexpr [reverse_iterator](#) [operator-](#) (difference_type __n) const
- constexpr [reverse_iterator](#) & [operator--](#) ()
- constexpr [reverse_iterator](#) [operator--](#) (int)
- constexpr [reverse_iterator](#) & [operator-=](#) (difference_type __n)
- constexpr pointer [operator->](#) () const
- [reverse_iterator](#) & [operator=](#) (const [reverse_iterator](#) &)=default
- template<typename _Iter >
requires __convertible<_Iter> && assignable_from<_Iterator&, const _Iter&>
constexpr [reverse_iterator](#) & [operator=](#) (const [reverse_iterator](#)< _Iter > &__x) noexcept(*/*conditional */*)
- constexpr reference [operator\[\]](#) (difference_type __n) const

Protected Types

- typedef [iterator_traits](#)<_Iterator> [__traits_type](#)

Protected Attributes

- _Iterator **current**

Friends

- constexpr iter_value_reference_t<_Iterator> **iter_move** (const [reverse_iterator](#) &__i) noexcept(is_nothrow_copy_constructible_v<_Iterator> &&noexcept(ranges::iter_move(--std::declval<_Iterator> &()))))
- template<indirectly_swappable<_Iterator> _Iter2>
constexpr void **iter_swap** (const [reverse_iterator](#) &__x, const [reverse_iterator](#)<_Iter2> &__y) noexcept(is_nothrow_copy_constructible_v<_Iterator> &&is_nothrow_copy_constructible_v<_Iter2> &&noexcept(ranges::iter_swap(--std::declval<_Iterator> &(), --std::declval<_Iter2> &()))))

6.900.1 Detailed Description

template<typename _Iterator>

class std::reverse_iterator<_Iterator>

Bidirectional and random access iterators have corresponding reverse iterator adaptors that iterate through the data structure in the opposite direction. They have the same signatures as the corresponding iterators. The fundamental relation between a reverse iterator and its corresponding iterator *i* is established by the identity:

```
*(reverse_iterator(i)) == *(i - 1)
```

This mapping is dictated by the fact that while there is always a pointer past the end of an array, there might not be a valid pointer before the beginning of an array. [24.4.1]/1,2

Reverse iterators can be tricky and surprising at first. Their semantics make sense, however, and the trickiness is a side effect of the requirement that the iterators must be safe.

6.900.2 Constructor & Destructor Documentation

reverse_iterator() [1/4]

```
template<typename _Iterator>
```

```
constexpr std::reverse_iterator<_Iterator>::reverse_iterator ( ) [inline], [constexpr], [noexcept]
```

The default constructor value-initializes member `current`. If it is a pointer, that means it is zero-initialized.

reverse_iterator() [2/4]

```
template<typename _Iterator>
```

```
constexpr std::reverse_iterator<_Iterator>::reverse_iterator (
    iterator_type __x ) [inline], [explicit], [constexpr], [noexcept]
```

This iterator will move in the opposite direction that `x` does.

reverse_iterator() [3/4]

```
template<typename _Iterator>
```

```
constexpr std::reverse_iterator<_Iterator>::reverse_iterator (
    const reverse_iterator<_Iterator> &__x ) [inline], [constexpr], [noexcept]
```

The copy constructor is normal.

reverse_iterator() [4/4]

```
template<typename _Iterator>
```

```
template<typename _Iter>
```

```
requires __convertible<_Iter>
constexpr std::reverse_iterator< _Iterator >::reverse_iterator (
    const reverse_iterator< _Iter > & __x ) [inline], [constexpr], [noexcept]
```

A reverse_iterator across other types can be copied if the underlying iterator can be converted to the type of current.

6.900.3 Member Function Documentation

base()

```
template<typename _Iterator >
constexpr iterator_type std::reverse_iterator< _Iterator >::base ( ) const [inline], [constexpr],
[noexcept]
```

Returns

current, the iterator used for underlying work.

Referenced by [std::operator==\(\)](#).

operator*()

```
template<typename _Iterator >
constexpr reference std::reverse_iterator< _Iterator >::operator* ( ) const [inline], [constexpr]
```

Returns

A reference to the value at --current

This requires that --current is dereferenceable.

Warning

This implementation requires that for an iterator of the underlying iterator type, x, a reference obtained by *x remains valid after x has been modified or destroyed. This is a bug: <http://gcc.gnu.org/PR51823>

operator+()

```
template<typename _Iterator >
constexpr reverse_iterator std::reverse_iterator< _Iterator >::operator+ (
    difference_type __n ) const [inline], [constexpr]
```

Returns

A reverse_iterator that refers to current - __n

The underlying iterator must be a Random Access Iterator.

operator++() [1/2]

```
template<typename _Iterator >
constexpr reverse_iterator & std::reverse_iterator< _Iterator >::operator++ ( ) [inline], [constexpr]
```

Returns

*this

Decrements the underlying iterator.

operator++() [2/2]

```
template<typename _Iterator >
constexpr reverse_iterator std::reverse_iterator< _Iterator >::operator++ (
    int ) [inline], [constexpr]
```

Returns

The original value of `*this`

Decrements the underlying iterator.

operator+=()

```
template<typename _Iterator >
constexpr reverse_iterator & std::reverse_iterator< _Iterator >::operator+= (
    difference_type __n ) [inline], [constexpr]
```

Returns

`*this`

Moves the underlying iterator backwards `__n` steps. The underlying iterator must be a Random Access Iterator.

operator-()

```
template<typename _Iterator >
constexpr reverse_iterator std::reverse_iterator< _Iterator >::operator- (
    difference_type __n ) const [inline], [constexpr]
```

Returns

A `reverse_iterator` that refers to `current - __n`

The underlying iterator must be a Random Access Iterator.

operator--() [1/2]

```
template<typename _Iterator >
constexpr reverse_iterator & std::reverse_iterator< _Iterator >::operator-- ( ) [inline], [constexpr]
```

Returns

`*this`

Increments the underlying iterator.

operator--() [2/2]

```
template<typename _Iterator >
constexpr reverse_iterator std::reverse_iterator< _Iterator >::operator-- (
    int ) [inline], [constexpr]
```

Returns

A `reverse_iterator` with the previous value of `*this`

Increments the underlying iterator.

operator-=()

```
template<typename _Iterator >
constexpr reverse_iterator & std::reverse_iterator< _Iterator >::operator-= (
    difference_type __n ) [inline], [constexpr]
```

Returns

*this

Moves the underlying iterator forwards `__n` steps. The underlying iterator must be a Random Access Iterator.

operator->()

```
template<typename _Iterator >
constexpr pointer std::reverse_iterator< _Iterator >::operator-> ( ) const [inline], [constexpr]
```

Returns

A pointer to the value at `--current`

This requires that `--current` is dereferenceable.

operator[]()

```
template<typename _Iterator >
constexpr reference std::reverse_iterator< _Iterator >::operator[] (
    difference_type __n ) const [inline], [constexpr]
```

Returns

The value at `current - __n - 1`

The underlying iterator must be a Random Access Iterator.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

6.901 __gnu_cxx::rope< _CharT, _Alloc > Class Template Reference

Inherits `__gnu_cxx::Rope_base< _CharT, _Alloc >`.

Public Types

- typedef `_Rope_RopeConcatenation< _CharT, _Alloc >` **__C**
- typedef `_Rope_RopeFunction< _CharT, _Alloc >` **__F**
- typedef `_Rope_RopeLeaf< _CharT, _Alloc >` **__L**
- typedef `_Rope_RopeSubstring< _CharT, _Alloc >` **__S**
- typedef `__alloc_traits< _Alloc >::template rebind< __C >::other` **_CAlloc**
- typedef `__alloc_traits< _Alloc >::template rebind< _CharT >::other` **_DataAlloc**
- typedef `__alloc_traits< _Alloc >::template rebind< __F >::other` **_FAlloc**
- typedef `__alloc_traits< _Alloc >::template rebind< __L >::other` **_LAlloc**
- typedef `__alloc_traits< _Alloc >::template rebind< __S >::other` **_SAlloc**
- typedef `_Rope_const_iterator< _CharT, _Alloc >` **const_iterator**
- typedef `const _CharT *` **const_pointer**
- typedef `_CharT` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `std::ptrdiff_t` **difference_type**

- typedef `_Rope_iterator<_CharT, _Alloc>` **iterator**
- typedef `_Rope_char_ptr_proxy<_CharT, _Alloc>` **pointer**
- typedef `_Rope_char_ref_proxy<_CharT, _Alloc>` **reference**
- typedef `std::reverse_iterator<iterator>` **reverse_iterator**
- typedef `std::size_t` **size_type**
- typedef `_CharT` **value_type**

Public Member Functions

- **rope** (`_CharT __c`, `const allocator_type &__a=allocator_type()`)
- **rope** (`char_producer<_CharT> *__fn`, `size_type __len`, `bool __delete_fn`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const _CharT *__s`, `const _CharT *__e`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const _CharT *__s`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const _CharT *__s`, `size_type __len`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const allocator_type &__a=allocator_type()`)
- **rope** (`const const_iterator &__s`, `const const_iterator &__e`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const iterator &__s`, `const iterator &__e`, `const allocator_type &__a=allocator_type()`)
- **rope** (`const rope &__x`, `const allocator_type &__a=allocator_type()`)
- **rope** (`size_type __n`, `_CharT __c`, `const allocator_type &__a=allocator_type()`)
- `allocator_type &_M_get_allocator ()`
- `const allocator_type &_M_get_allocator () const`
- **rope** & **append** ()
- **rope** & **append** (`_CharT __c`)
- **rope** & **append** (`const _CharT *__c_string`)
- **rope** & **append** (`const _CharT *__iter`, `size_type __n`)
- **rope** & **append** (`const _CharT *__s`, `const _CharT *__e`)
- **rope** & **append** (`const rope &__y`)
- **rope** & **append** (`const_iterator __s`, `const_iterator __e`)
- **rope** & **append** (`size_type __n`, `_CharT __c`)
- `void apply_to_pieces` (`size_type __begin`, `size_type __end`, `_Rope_char_consumer<_CharT> &__c`) `const`
- `_CharT at` (`size_type __pos`) `const`
- `_CharT back` () `const`
- `void balance` ()
- `const_iterator begin` ()
- `const_iterator begin` () `const`
- `const _CharT * c_str` () `const`
- `void clear` ()
- `int compare` (`const rope &__y`) `const`
- `const_iterator const_begin` () `const`
- `const_iterator const_end` () `const`
- `const_reverse_iterator const_rbegin` () `const`
- `const_reverse_iterator const_rend` () `const`
- `void copy` (`_CharT *__buffer`) `const`
- `size_type copy` (`size_type __pos`, `size_type __n`, `_CharT *__buffer`) `const`
- `void delete_c_str` ()
- `void dump` ()
- `bool empty` () `const`
- `const_iterator end` ()
- `const_iterator end` () `const`
- `iterator erase` (`const iterator &__p`)

- iterator **erase** (const iterator &__p, const iterator &__q)
- void **erase** (size_type __p, size_type __n)
- size_type **find** (_CharT __c, size_type __pos=0) const
- size_type **find** (const _CharT *__s, size_type __pos=0) const
- _CharT **front** () const
- allocator_type **get_allocator** () const
- iterator **insert** (const iterator &__p)
- iterator **insert** (const iterator &__p, _CharT __c)
- iterator **insert** (const iterator &__p, const _CharT *__i, const _CharT *__j)
- iterator **insert** (const iterator &__p, const _CharT *__i, size_type __n)
- iterator **insert** (const iterator &__p, const _CharT *c_string)
- iterator **insert** (const iterator &__p, const const_iterator &__i, const const_iterator &__j)
- iterator **insert** (const iterator &__p, const iterator &__i, const iterator &__j)
- iterator **insert** (const iterator &__p, const [rope](#) &__r)
- iterator **insert** (const iterator &__p, size_type __n, _CharT __c)
- void **insert** (size_type __p)
- void **insert** (size_type __p, _CharT __c)
- void **insert** (size_type __p, const _CharT *c_string)
- void **insert** (size_type __p, const _CharT *__i, const _CharT *__j)
- void **insert** (size_type __p, const _CharT *__i, size_type __n)
- void **insert** (size_type __p, const const_iterator &__i, const const_iterator &__j)
- void **insert** (size_type __p, const iterator &__i, const iterator &__j)
- void **insert** (size_type __p, const [rope](#) &__r)
- void **insert** (size_type __p, size_type __n, _CharT __c)
- size_type **length** () const
- size_type **max_size** () const
- iterator **mutable_begin** ()
- iterator **mutable_end** ()
- [reverse_iterator](#) **mutable_rbegin** ()
- reference **mutable_reference_at** (size_type __pos)
- [reverse_iterator](#) **mutable_rend** ()
- [rope](#) & **operator=** (const [rope](#) &__x)
- _CharT **operator[]** (size_type __pos) const
- void **pop_back** ()
- void **pop_front** ()
- void **push_back** (_CharT __x)
- void **push_front** (_CharT __x)
- [const_reverse_iterator](#) **rbegin** ()
- [const_reverse_iterator](#) **rbegin** () const
- [const_reverse_iterator](#) **rend** ()
- [const_reverse_iterator](#) **rend** () const
- void **replace** (const iterator &__p, _CharT __c)
- void **replace** (const iterator &__p, const _CharT *c_string)
- void **replace** (const iterator &__p, const _CharT *__i, const _CharT *__j)
- void **replace** (const iterator &__p, const _CharT *__i, size_type __n)
- void **replace** (const iterator &__p, const iterator &__q, _CharT __c)
- void **replace** (const iterator &__p, const iterator &__q, const _CharT *c_string)
- void **replace** (const iterator &__p, const iterator &__q, const _CharT *__i, const _CharT *__j)
- void **replace** (const iterator &__p, const iterator &__q, const _CharT *__i, size_type __n)
- void **replace** (const iterator &__p, const iterator &__q, const const_iterator &__i, const const_iterator &__j)
- void **replace** (const iterator &__p, const iterator &__q, const iterator &__i, const iterator &__j)

- void **replace** (const iterator &__p, const iterator &__q, const [rope](#) &__r)
- void **replace** (const iterator &__p, const [rope](#) &__r)
- void **replace** (const iterator &__p, const_iterator __i, const_iterator __j)
- void **replace** (const iterator &__p, iterator __i, iterator __j)
- void **replace** (size_type __p, _CharT __c)
- void **replace** (size_type __p, const _CharT *__c_string)
- void **replace** (size_type __p, const _CharT *__i, const _CharT *__j)
- void **replace** (size_type __p, const _CharT *__i, size_type __i_len)
- void **replace** (size_type __p, const const_iterator &__i, const const_iterator &__j)
- void **replace** (size_type __p, const iterator &__i, const iterator &__j)
- void **replace** (size_type __p, const [rope](#) &__r)
- void **replace** (size_type __p, size_type __n, _CharT __c)
- void **replace** (size_type __p, size_type __n, const _CharT *__c_string)
- void **replace** (size_type __p, size_type __n, const _CharT *__i, const _CharT *__j)
- void **replace** (size_type __p, size_type __n, const _CharT *__i, size_type __i_len)
- void **replace** (size_type __p, size_type __n, const const_iterator &__i, const const_iterator &__j)
- void **replace** (size_type __p, size_type __n, const iterator &__i, const iterator &__j)
- void **replace** (size_type __p, size_type __n, const [rope](#) &__r)
- const _CharT * **replace_with_c_str** ()
- size_type **size** () const
- [rope](#)<_CharT, _Alloc> **substr** (const_iterator __start)
- [rope](#) **substr** (const_iterator __start, const_iterator __end) const
- [rope](#) **substr** (iterator __start) const
- [rope](#) **substr** (iterator __start, iterator __end) const
- [rope](#) **substr** (size_type __start, size_type __len=1) const
- void **swap** ([rope](#) &__b)

Static Public Member Functions

- static __C * **_C_allocate** (std::size_t __n)
- static void **_C_deallocate** (__C *__p, std::size_t __n)
- static _CharT * **_Data_allocate** (std::size_t __n)
- static void **_Data_deallocate** (_CharT *__p, std::size_t __n)
- static __F * **_F_allocate** (std::size_t __n)
- static void **_F_deallocate** (__F *__p, std::size_t __n)
- static __L * **_L_allocate** (std::size_t __n)
- static void **_L_deallocate** (__L *__p, std::size_t __n)
- static __S * **_S_allocate** (std::size_t __n)
- static void **_S_deallocate** (__S *__p, std::size_t __n)

Public Attributes

- _RopeRep * **_M_tree_ptr**

Static Public Attributes

- static const size_type **npos**

Protected Types

- enum { **_S_copy_max** }
- typedef _Rope_base< _CharT, _Alloc > **_Base**
- typedef _CharT * **_Cstrptr**
- typedef _Rope_RopeConcatenation< _CharT, _Alloc > **_RopeConcatenation**
- typedef _Rope_RopeFunction< _CharT, _Alloc > **_RopeFunction**
- typedef _Rope_RopeLeaf< _CharT, _Alloc > **_RopeLeaf**
- typedef _Rope_RopeRep< _CharT, _Alloc > **_RopeRep**
- typedef _Rope_RopeSubstring< _CharT, _Alloc > **_RopeSubstring**
- typedef _Rope_self_destruct_ptr< _CharT, _Alloc > **_Self_destruct_ptr**
- typedef _Base::allocator_type **allocator_type**

Static Protected Member Functions

- static size_type **_S_allocated_capacity** (size_type __n)
- static bool **_S_apply_to_pieces** (_Rope_char_consumer< _CharT > &__c, const _RopeRep *__r, size_type __begin, size_type __end)
- static _RopeRep * **_S_concat** (_RopeRep *__left, _RopeRep *__right)
- static _RopeRep * **_S_concat_char_iter** (_RopeRep *__r, const _CharT *__iter, size_type __slen, allocator_type &__a)
- static _RopeRep * **_S_destr_concat_char_iter** (_RopeRep *__r, const _CharT *__iter, size_type __slen, allocator_type &__a)
- static _RopeLeaf * **_S_destr_leaf_concat_char_iter** (_RopeLeaf *__r, const _CharT *__iter, size_type __slen)
- static _CharT **_S_fetch** (_RopeRep *__r, size_type __pos)
- static _CharT * **_S_fetch_ptr** (_RopeRep *__r, size_type __pos)
- static bool **_S_is0** (_CharT __c)
- static _RopeLeaf * **_S_leaf_concat_char_iter** (_RopeLeaf *__r, const _CharT *__iter, size_type __slen)
- static _RopeConcatenation * **_S_new_RopeConcatenation** (_RopeRep *__left, _RopeRep *__right, allocator_type &__a)
- static _RopeFunction * **_S_new_RopeFunction** (char_producer< _CharT > *__f, size_type __size, bool __d, allocator_type &__a)
- static _RopeLeaf * **_S_new_RopeLeaf** (_CharT *__s, size_type __size, allocator_type &__a)
- static _RopeSubstring * **_S_new_RopeSubstring** (_Rope_RopeRep< _CharT, _Alloc > *__b, size_type __s, size_type __l, allocator_type &__a)
- static void **_S_ref** (_RopeRep *__t)
- static _RopeLeaf * **_S_RopeLeaf_from_unowned_char_ptr** (const _CharT *__s, size_type __size, allocator_type &__a)
- static size_type **_S_rounded_up_size** (size_type __n)
- static _RopeRep * **_S_substring** (_RopeRep *__base, size_type __start, size_type __endp1)
- static _RopeRep * **_S_tree_concat** (_RopeRep *__left, _RopeRep *__right)
- static void **_S_unref** (_RopeRep *__t)
- static _RopeRep * **replace** (_RopeRep *__old, size_type __pos1, size_type __pos2, _RopeRep *__r)

Static Protected Attributes

- static _CharT **_S_empty_c_str** [1]

Friends

- class `_Rope_char_ptr_proxy<_CharT, _Alloc>`
- class `_Rope_char_ref_proxy<_CharT, _Alloc>`
- class `_Rope_const_iterator<_CharT, _Alloc>`
- class `_Rope_iterator<_CharT, _Alloc>`
- class `_Rope_iterator_base<_CharT, _Alloc>`
- struct `_Rope_RopeRep<_CharT, _Alloc>`
- struct `_Rope_RopeSubstring<_CharT, _Alloc>`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, _CharT2 __right)`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, const _CharT2 *__right)`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, const rope<_CharT2, _Alloc2> &__right)`

6.901.1 Detailed Description

template<class `_CharT`, class `_Alloc`>
class `__gnu_cxx::rope<_CharT, _Alloc>`

This is an SGI extension.

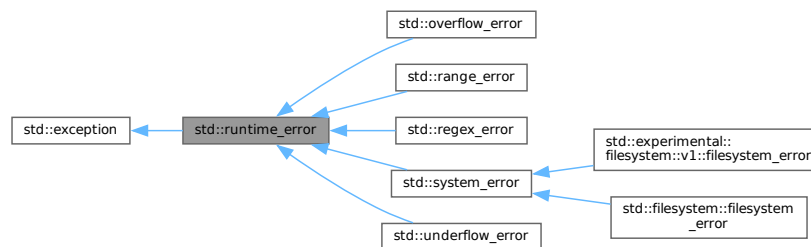
Todo Needs documentation! See <http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation-style.html>

The documentation for this class was generated from the following files:

- `rope`
- `ropeimpl.h`

6.902 std::runtime_error Class Reference

Inheritance diagram for `std::runtime_error`:



Public Member Functions

- `runtime_error (const char *) _GLIBCXX_TXN_SAFE`
- `runtime_error (const runtime_error &)=default`
- `runtime_error (const string &__arg) _GLIBCXX_TXN_SAFE`
- `runtime_error (runtime_error &&) noexcept`

- [runtime_error](#) & **operator=** (const [runtime_error](#) &)=default
- [runtime_error](#) & **operator=** ([runtime_error](#) &&) noexcept
- virtual const char * [what](#) () const noexcept

6.902.1 Detailed Description

One of two subclasses of exception.

Runtime errors represent problems outside the scope of a program; they cannot be easily predicted and can generally only be caught as the program executes.

6.902.2 Constructor & Destructor Documentation

runtime_error()

```
std::runtime_error::runtime_error (  
    const string & __arg ) [explicit]
```

Takes a character string describing the error.

6.902.3 Member Function Documentation

what()

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

6.903 [__gnu_pbds::sample_probe_fn](#) Class Reference

```
#include <sample_probe_fn.hpp>
```

Public Types

- typedef std::size_t **size_type**

Public Member Functions

- [sample_probe_fn](#) ()
- [sample_probe_fn](#) (const [sample_probe_fn](#) &)
- void [swap](#) ([sample_probe_fn](#) &)

Protected Member Functions

- size_type [operator\(\)](#) (key_const_reference r_key, size_type i) const

6.903.1 Detailed Description

A sample probe policy.

6.903.2 Constructor & Destructor Documentation

sample_probe_fn() [1/2]

`__gnu_pbds::sample_probe_fn::sample_probe_fn ()`
Default constructor.

sample_probe_fn() [2/2]

`__gnu_pbds::sample_probe_fn::sample_probe_fn (`
 `const sample_probe_fn &)`
Copy constructor.

6.903.3 Member Function Documentation

operator>()

`size_type __gnu_pbds::sample_probe_fn::operator() (`
 `key_const_reference r_key,`
 `size_type i) const [inline], [protected]`
Returns the i-th offset from the hash value of some key r_key.

swap()

`void __gnu_pbds::sample_probe_fn::swap (`
 `sample_probe_fn &) [inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_probe_fn.hpp](#)

6.904 __gnu_pbds::sample_range_hashing Class Reference

```
#include <sample_range_hashing.hpp>
```

Public Types

- `typedef std::size_t size_type`

Public Member Functions

- [sample_range_hashing](#) ()
- [sample_range_hashing](#) (const [sample_range_hashing](#) &other)
- void [swap](#) ([sample_range_hashing](#) &other)

Protected Member Functions

- void [notify_resized](#) ([size_type](#))
- [size_type](#) [operator\(\)](#) ([size_type](#)) const

6.904.1 Detailed Description

A sample range-hashing functor.

6.904.2 Member Typedef Documentation

size_type

`typedef std::size_t __gnu_pbds::sample_range_hashing::size_type`
Size type.

6.904.3 Constructor & Destructor Documentation

sample_range_hashing() [1/2]

`__gnu_pbds::sample_range_hashing::sample_range_hashing ()`
Default constructor.

sample_range_hashing() [2/2]

`__gnu_pbds::sample_range_hashing::sample_range_hashing (`
 `const sample_range_hashing & other)`
Copy constructor.

6.904.4 Member Function Documentation

notify_resized()

`void __gnu_pbds::sample_range_hashing::notify_resized (`
 `size_type) [protected]`
Notifies the policy object that the container's size has changed to argument's size.

operator>()

`size_type __gnu_pbds::sample_range_hashing::operator() (`
 `size_type) const [inline], [protected]`
Transforms the `__hash` value hash into a ranged-hash value.

swap()

`void __gnu_pbds::sample_range_hashing::swap (`
 `sample_range_hashing & other) [inline]`
Swaps content.

The documentation for this class was generated from the following file:

- [sample_range_hashing.hpp](#)

6.905 __gnu_pbds::sample_ranged_hash_fn Class Reference

`#include <sample_ranged_hash_fn.hpp>`

Public Types

- `typedef std::size_t size_type`

Public Member Functions

- [sample_ranged_hash_fn \(\)](#)
- [sample_ranged_hash_fn \(const sample_ranged_hash_fn &\)](#)
- `void swap (sample_ranged_hash_fn &)`

Protected Member Functions

- void [notify_resized](#) (size_type)
- size_type [operator\(\)](#) (key_const_reference) const

6.905.1 Detailed Description

A sample ranged-hash functor.

6.905.2 Constructor & Destructor Documentation**`sample_ranged_hash_fn()` [1/2]**

```
__gnu_pbds::sample_ranged_hash_fn::sample_ranged_hash_fn ( )
```

Default constructor.

`sample_ranged_hash_fn()` [2/2]

```
__gnu_pbds::sample_ranged_hash_fn::sample_ranged_hash_fn (
    const sample\_ranged\_hash\_fn & )
```

Copy constructor.

6.905.3 Member Function Documentation**`notify_resized()`**

```
void __gnu_pbds::sample_ranged_hash_fn::notify_resized (
    size_type ) [protected]
```

Notifies the policy object that the container's `__size` has changed to `size`.

`operator>()`

```
size_type __gnu_pbds::sample_ranged_hash_fn::operator() (
    key_const_reference ) const [inline], [protected]
```

Transforms `key_const_reference` into a position within the table.

`swap()`

```
void __gnu_pbds::sample_ranged_hash_fn::swap (
    sample\_ranged\_hash\_fn & ) [inline]
```

Swaps content.

The documentation for this class was generated from the following file:

- [sample_ranged_hash_fn.hpp](#)

6.906 `__gnu_pbds::sample_ranged_probe_fn` Class Reference

```
#include <sample_ranged_probe_fn.hpp>
```

Public Types

- typedef std::size_t **size_type**

Public Member Functions

- **`sample_ranged_probe_fn`** (const [sample_ranged_probe_fn](#) &)
- void **`swap`** ([sample_ranged_probe_fn](#) &)

Protected Member Functions

- void **notify_resized** (size_type)
- size_type **operator()** (key_const_reference, std::size_t, size_type) const

6.906.1 Detailed Description

A sample ranged-probe functor.

The documentation for this class was generated from the following file:

- [sample_ranged_probe_fn.hpp](#)

6.907 __gnu_pbds::sample_resize_policy Class Reference

```
#include <sample_resize_policy.hpp>
```

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_resize_policy](#) ()
- [sample_range_hashing](#) (const [sample_resize_policy](#) &other)
- void [swap](#) ([sample_resize_policy](#) &other)

Protected Member Functions

- [size_type](#) [get_new_size](#) ([size_type](#) size, [size_type](#) num_used_e) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) ([size_type](#) num_e)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) ([size_type](#) num_e)
- void [notify_resized](#) ([size_type](#) new_size)

6.907.1 Detailed Description

A sample resize policy.

6.907.2 Member Typedef Documentation

size_type

```
typedef std::size_t __gnu_pbds::sample_resize_policy::size_type
```

Size type.

6.907.3 Constructor & Destructor Documentation

sample_resize_policy()

```
__gnu_pbds::sample_resize_policy::sample_resize_policy ( )
```

Default constructor.

6.907.4 Member Function Documentation

get_new_size()

```
size_type __gnu_pbds::sample_resize_policy::get_new_size (
    size_type size,
    size_type num_used_e ) const [protected]
```

Queries what the new size should be.

is_resize_needed()

```
bool __gnu_pbds::sample_resize_policy::is_resize_needed ( ) const [inline], [protected]
```

Queries whether a resize is needed.

notify_cleared()

```
void __gnu_pbds::sample_resize_policy::notify_cleared ( ) [protected]
```

Notifies the table was cleared.

notify_erase_search_collision()

```
void __gnu_pbds::sample_resize_policy::notify_erase_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_erase_search_end()

```
void __gnu_pbds::sample_resize_policy::notify_erase_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_erase_search_start()

```
void __gnu_pbds::sample_resize_policy::notify_erase_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_erased()

```
void __gnu_pbds::sample_resize_policy::notify_erased (
    size_type num_e ) [inline], [protected]
```

Notifies an element was erased.

notify_find_search_collision()

```
void __gnu_pbds::sample_resize_policy::notify_find_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_find_search_end()

```
void __gnu_pbds::sample_resize_policy::notify_find_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_find_search_start()

```
void __gnu_pbds::sample_resize_policy::notify_find_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_insert_search_collision()

```
void __gnu_pbds::sample_resize_policy::notify_insert_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_insert_search_end()

```
void __gnu_pbds::sample_resize_policy::notify_insert_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_insert_search_start()

```
void __gnu_pbds::sample_resize_policy::notify_insert_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_inserted()

```
void __gnu_pbds::sample_resize_policy::notify_inserted (
    size_type num_e ) [inline], [protected]
```

Notifies an element was inserted.

notify_resized()

```
void __gnu_pbds::sample_resize_policy::notify_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized to new_size.

sample_range_hashing()

```
__gnu_pbds::sample_resize_policy::sample_range_hashing (
    const sample_resize_policy & other )
```

Copy constructor.

swap()

```
void __gnu_pbds::sample_resize_policy::swap (
    sample_resize_policy & other ) [inline]
```

Swaps content.

The documentation for this class was generated from the following file:

- [sample_resize_policy.hpp](#)

6.908 __gnu_pbds::sample_resize_trigger Class Reference

```
#include <sample_resize_trigger.hpp>
```

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_resize_trigger](#) ()
- [sample_range_hashing](#) (const [sample_resize_trigger](#) &)
- void [swap](#) ([sample_resize_trigger](#) &)

Protected Member Functions

- bool [is_grow_needed](#) ([size_type](#) size, [size_type](#) num_entries) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) ([size_type](#) num_entries)
- void [notify_externally_resized](#) ([size_type](#) new_size)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) ([size_type](#) num_entries)
- void [notify_resized](#) ([size_type](#) new_size)

6.908.1 Detailed Description

A sample resize trigger policy.

6.908.2 Member Typedef Documentation

size_type

typedef std::size_t __gnu_pbds::sample_resize_trigger::size_type
Size type.

6.908.3 Constructor & Destructor Documentation

sample_resize_trigger()

`__gnu_pbds::sample_resize_trigger::sample_resize_trigger ()`
Default constructor.

6.908.4 Member Function Documentation

is_grow_needed()

```
bool __gnu_pbds::sample_resize_trigger::is_grow_needed (
    size\_type size,
    size\_type num_entries ) const [inline], [protected]
```

Queries whether a grow is needed.

is_resize_needed()

```
bool __gnu_pbds::sample_resize_trigger::is_resize_needed ( ) const [inline], [protected]
```

Queries whether a resize is needed.

notify_cleared()

```
void __gnu_pbds::sample_resize_trigger::notify_cleared ( ) [protected]
```

Notifies the table was cleared.

notify_erase_search_collision()

```
void __gnu_pbds::sample_resize_trigger::notify_erase_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_erase_search_end()

```
void __gnu_pbds::sample_resize_trigger::notify_erase_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_erase_search_start()

```
void __gnu_pbds::sample_resize_trigger::notify_erase_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_erased()

```
void __gnu_pbds::sample_resize_trigger::notify_erased (
    size_type num_entries ) [inline], [protected]
```

Notifies an element was erased.

notify_externally_resized()

```
void __gnu_pbds::sample_resize_trigger::notify_externally_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized externally.

notify_find_search_collision()

```
void __gnu_pbds::sample_resize_trigger::notify_find_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_find_search_end()

```
void __gnu_pbds::sample_resize_trigger::notify_find_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_find_search_start()

```
void __gnu_pbds::sample_resize_trigger::notify_find_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_insert_search_collision()

```
void __gnu_pbds::sample_resize_trigger::notify_insert_search_collision ( ) [inline], [protected]
```

Notifies a search encountered a collision.

notify_insert_search_end()

```
void __gnu_pbds::sample_resize_trigger::notify_insert_search_end ( ) [inline], [protected]
```

Notifies a search ended.

notify_insert_search_start()

```
void __gnu_pbds::sample_resize_trigger::notify_insert_search_start ( ) [inline], [protected]
```

Notifies a search started.

notify_inserted()

```
void __gnu_pbds::sample_resize_trigger::notify_inserted (
    size_type num_entries ) [inline], [protected]
```

Notifies an element was inserted. the total number of entries in the table is num_entries.

notify_resized()

```
void __gnu_pbds::sample_resize_trigger::notify_resized (
    size_type new_size ) [protected]
```

Notifies the table was resized as a result of this object's signifying that a resize is needed.

sample_range_hashing()

```
__gnu_pbds::sample_resize_trigger::sample_range_hashing (
    const sample_resize_trigger & )
```

Copy constructor.

swap()

```
void __gnu_pbds::sample_resize_trigger::swap (
    sample_resize_trigger & ) [inline]
```

Swaps content.
The documentation for this class was generated from the following file:

- [sample_resize_trigger.hpp](#)

6.909 `__gnu_pbds::sample_size_policy` Class Reference

```
#include <sample_size_policy.hpp>
```

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_size_policy](#) ()
- [sample_range_hashing](#) (const [sample_size_policy](#) &)
- void [swap](#) ([sample_size_policy](#) &other)

Protected Member Functions

- [size_type](#) [get_nearest_larger_size](#) ([size_type](#) size) const
- [size_type](#) [get_nearest_smaller_size](#) ([size_type](#) size) const

6.909.1 Detailed Description

A sample size policy.

6.909.2 Member Typedef Documentation

size_type

`typedef std::size_t __gnu_pbds::sample_size_policy::size_type`
Size type.

6.909.3 Constructor & Destructor Documentation

sample_size_policy()

`__gnu_pbds::sample_size_policy::sample_size_policy ()`
Default constructor.

6.909.4 Member Function Documentation

get_nearest_larger_size()

`size_type __gnu_pbds::sample_size_policy::get_nearest_larger_size (`
`size_type size) const [inline], [protected]`

Given a `__size` size, returns a `__size` that is larger.

get_nearest_smaller_size()

`size_type __gnu_pbds::sample_size_policy::get_nearest_smaller_size (`
`size_type size) const [inline], [protected]`

Given a `__size` size, returns a `__size` that is smaller.

sample_range_hashing()

`__gnu_pbds::sample_size_policy::sample_range_hashing (`
`const sample_size_policy &)`

Copy constructor.

swap()

`void __gnu_pbds::sample_size_policy::swap (`
`sample_size_policy & other) [inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_size_policy.hpp](#)

6.910 __gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc > Class Template Reference

`#include <sample_tree_node_update.hpp>`

6.910.1 Detailed Description

`template<typename Const_Node_Iter, typename Node_Iter, typename Cmp_Fn, typename _Alloc>`
`class __gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >`

A sample node updatator.

The documentation for this class was generated from the following file:

- [sample_tree_node_update.hpp](#)

6.911 `__gnu_pbds::sample_trie_access_traits` Struct Reference

```
#include <sample_trie_access_traits.hpp>
```

Public Types

- enum { `max_size` }
- typedef `std::string::const_iterator` `const_iterator`
- typedef char `e_type`
- typedef `rebind_traits< _Alloc, key_type >::const_reference` `key_const_reference`
- typedef `std::string` `key_type`
- typedef `std::size_t` `size_type`

Static Public Member Functions

- static `const_iterator` `begin` (`key_const_reference`)
- static `size_type` `e_pos` (`e_type`)
- static `const_iterator` `end` (`key_const_reference`)

6.911.1 Detailed Description

A sample trie element access traits.

6.911.2 Member Typedef Documentation

`e_type`

```
typedef char __gnu_pbds::sample_trie_access_traits::e_type
```

Element type.

6.911.3 Member Function Documentation

`begin()`

```
static const_iterator __gnu_pbds::sample_trie_access_traits::begin (
    key_const_reference ) [inline], [static]
```

Returns a `const_iterator` to the first element of `r_key`.

`e_pos()`

```
static size_type __gnu_pbds::sample_trie_access_traits::e_pos (
    e_type ) [inline], [static]
```

Maps an element to a position.

`end()`

```
static const_iterator __gnu_pbds::sample_trie_access_traits::end (
    key_const_reference ) [inline], [static]
```

Returns a `const_iterator` to the after-last element of `r_key`.

The documentation for this struct was generated from the following file:

- [sample_trie_access_traits.hpp](#)

6.912 `__gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Ltr, _ATraits, _Alloc >` Class Template Reference

```
#include <sample_trie_node_update.hpp>
```

Public Types

- `typedef std::size_t metadata_type`

Protected Member Functions

- [sample_trie_node_update](#) ()
- `void operator()` (node_iterator, node_const_iterator) const

6.912.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

A sample node updatator.

6.912.2 Constructor & Destructor Documentation

`sample_trie_node_update()`

```
template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
__gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::sample_trie_node_↵
update ( ) [protected]
Default constructor.
```

6.912.3 Member Function Documentation

`operator>()()`

```
template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
void __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::operator() (
    node_iterator ,
    node_const_iterator ) const [inline], [protected]
```

Updates the rank of a node through a node_iterator node_it; end_nd_it is the end node iterator.

The documentation for this class was generated from the following file:

- [sample_trie_node_update.hpp](#)

6.913 __gnu_pbds::sample_update_policy Struct Reference

```
#include <sample_update_policy.hpp>
```

Public Member Functions

- [sample_update_policy](#) ()
- [sample_update_policy](#) (const [sample_update_policy](#) &)
- `void swap` ([sample_update_policy](#) &other)

Protected Types

- `typedef some_metadata_type metadata_type`

Protected Member Functions

- [metadata_type operator\(\)](#) () const
- `bool operator()` (metadata_reference) const

6.913.1 Detailed Description

A sample list-update policy.

6.913.2 Member Typedef Documentation

metadata_type

```
typedef some_metadata_type __gnu_pbds::sample_update_policy::metadata_type [protected]
```

Metadata on which this functor operates.

6.913.3 Constructor & Destructor Documentation

sample_update_policy() [1/2]

```
__gnu_pbds::sample_update_policy::sample_update_policy ( )
```

Default constructor.

sample_update_policy() [2/2]

```
__gnu_pbds::sample_update_policy::sample_update_policy (
    const sample_update_policy & )
```

Copy constructor.

6.913.4 Member Function Documentation

operator>() [1/2]

```
metadata_type __gnu_pbds::sample_update_policy::operator() ( ) const [protected]
```

Creates a metadata object.

operator>() [2/2]

```
bool __gnu_pbds::sample_update_policy::operator() (
    metadata_reference ) const [protected]
```

Decides whether a metadata object should be moved to the front of the list. A list-update based containers object will call this method to decide whether to move a node to the front of the list. The method should return true if the node should be moved to the front of the list.

swap()

```
void __gnu_pbds::sample_update_policy::swap (
    sample_update_policy & other ) [inline]
```

Swaps content.

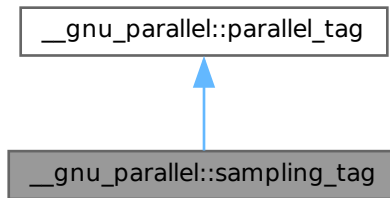
The documentation for this struct was generated from the following file:

- [sample_update_policy.hpp](#)

6.914 __gnu_parallel::sampling_tag Struct Reference

```
#include <tags.h>
```


Inheritance diagram for `__gnu_parallel::sampling_tag`:



Public Member Functions

- **sampling_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

6.914.1 Detailed Description

Forces parallel merging with exact splitting, at compile time.

6.914.2 Member Function Documentation

`__get_num_threads()`

[_ThreadIndex](#) `__gnu_parallel::parallel_tag::__get_num_threads ()` [inline], [inherited]

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort()`, and `__gnu_parallel::__parallel_sort()`.

`set_num_threads()`

void `__gnu_parallel::parallel_tag::set_num_threads (`
 [_ThreadIndex](#) __num_threads) [inline], [inherited]

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.915 std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs > Class Template Reference

Inherits `_OuterAlloc`.

Public Types

- typedef `__traits::const_pointer` **const_pointer**
- typedef `__traits::const_void_pointer` **const_void_pointer**
- typedef `__traits::difference_type` **difference_type**
- typedef `__inner_type::__type` **inner_allocator_type**
- typedef `__and< typename __traits::is_always_equal, typename allocator_traits< _InnerAllocs >::is_always_equal... >::type` **is_always_equal**
- typedef `_OuterAlloc` **outer_allocator_type**
- typedef `__traits::pointer` **pointer**
- typedef `__or< typename __traits::propagate_on_container_copy_assignment, typename allocator_traits< _InnerAllocs >::propagate_on_container_copy_assignment... >::type` **propagate_on_container_copy_assignment**
- typedef `__or< typename __traits::propagate_on_container_move_assignment, typename allocator_traits< _InnerAllocs >::propagate_on_container_move_assignment... >::type` **propagate_on_container_move_assignment**
- typedef `__or< typename __traits::propagate_on_container_swap, typename allocator_traits< _InnerAllocs >::propagate_on_container_swap... >::type` **propagate_on_container_swap**
- typedef `__traits::size_type` **size_type**
- typedef `__traits::value_type` **value_type**
- typedef `__traits::void_pointer` **void_pointer**

Public Member Functions

- template<typename _Outer2, typename = _Constructible<_Outer2>>
 scoped_allocator_adaptor (_Outer2 &&__outer, const _InnerAllocs &... __inner)
- **scoped_allocator_adaptor** (const `scoped_allocator_adaptor` &__other)
- template<typename _Outer2, typename = _Constructible<const _Outer2&>>
 scoped_allocator_adaptor (const `scoped_allocator_adaptor` <_Outer2, _InnerAllocs... > &__other)
- **scoped_allocator_adaptor** (`scoped_allocator_adaptor` &&__other)
- template<typename _Outer2, typename = _Constructible<_Outer2>>
 scoped_allocator_adaptor (`scoped_allocator_adaptor` <_Outer2, _InnerAllocs... > &&__other)
- pointer **allocate** (size_type __n)
- pointer **allocate** (size_type __n, const_void_pointer __hint)
- template<typename _Tp, typename... _Args>
 void **construct** (_Tp *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- template<typename _Tp >
 void **destroy** (_Tp *__p)
- const inner_allocator_type & **inner_allocator** () const noexcept
- inner_allocator_type & **inner_allocator** () noexcept
- size_type **max_size** () const
- `scoped_allocator_adaptor` & **operator=** (const `scoped_allocator_adaptor` &)=default
- `scoped_allocator_adaptor` & **operator=** (`scoped_allocator_adaptor` &&)=default
- const outer_allocator_type & **outer_allocator** () const noexcept
- outer_allocator_type & **outer_allocator** () noexcept
- `scoped_allocator_adaptor` **select_on_container_copy_construction** () const

Friends

- `template<typename... >`
`class __inner_type_impl`
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
`bool operator== (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_adaptor< _OutA2, _InA... > &__b) noexcept`

Related Symbols

(Note that these are not member symbols.)

- `template<typename _OutA1, typename _OutA2, typename... _InA>`
`bool operator== (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_adaptor< _OutA2, _InA... > &__b) noexcept`

6.915.1 Detailed Description

```
template<typename _OuterAlloc, typename... _InnerAllocs>
class std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs >
```

An adaptor to recursively pass an allocator to the objects it constructs.
The documentation for this class was generated from the following file:

- [scoped_allocator](#)

6.916 `std::scoped_lock< _MutexTypes >` Class Template Reference

Public Member Functions

- `scoped_lock (_MutexTypes &... __m)`
- `scoped_lock (adopt_lock_t, _MutexTypes &... __m) noexcept`
- `scoped_lock (const scoped_lock &)=delete`
- `scoped_lock & operator= (const scoped_lock &)=delete`

6.916.1 Detailed Description

```
template<typename... _MutexTypes>
class std::scoped_lock< _MutexTypes >
```

A scoped lock type for multiple lockable objects.

A `scoped_lock` controls mutex ownership within a scope, releasing ownership in the destructor.

The documentation for this class was generated from the following file:

- [mutex](#)

6.917 `std::seed_seq` Class Reference

```
#include <random.h>
```

Public Types

- `typedef uint_least32_t result_type`

Public Member Functions

- [seed_seq](#) () noexcept
- `template<typename _InputIterator >`
`seed_seq` (_InputIterator __begin, _InputIterator __end)
- `seed_seq` (const [seed_seq](#) &)=delete
- `template<typename _IntType, typename = _Require<is_integral<_IntType>>>`
`seed_seq` (std::initializer_list<_IntType> __il)
- `template<typename _RandomAccessIterator >`
void **generate** (_RandomAccessIterator __begin, _RandomAccessIterator __end)
- [seed_seq](#) & **operator=** (const [seed_seq](#) &)=delete
- `template<typename _OutputIterator >`
void **param** (_OutputIterator __dest) const
- `size_t size` () const noexcept

6.917.1 Detailed Description

The `seed_seq` class generates sequences of seeds for random number generators.

6.917.2 Member Typedef Documentation**result_type**

```
typedef uint_least32_t std::seed\_seq::result\_type
```

The type of the seed vales.

6.917.3 Constructor & Destructor Documentation**seed_seq()**

```
std::seed_seq::seed_seq ( ) [inline], [noexcept]
```

Default constructor.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.918 `__gnu_cxx::select1st<_Pair>` Struct Template Reference

Inherits `std::_Select1st<_Pair>`.

Public Types

- `typedef _Pair` [argument_type](#)
- `typedef _Pair::first_type` [result_type](#)

Public Member Functions

- `_Pair::first_type & operator()` (_Pair &__x) const
- `template<typename _Pair2 >`
`_Pair2::first_type & operator()` (_Pair2 &__x) const
- `const _Pair::first_type & operator()` (const _Pair &__x) const
- `template<typename _Pair2 >`
`const _Pair2::first_type & operator()` (const _Pair2 &__x) const

6.918.1 Detailed Description

```
template<class _Pair>
struct __gnu_cxx::select1st< _Pair >
```

An [SGI extension](#) .

6.918.2 Member Typedef Documentation

argument_type

```
typedef _Pair std::unary_function< _Pair , _Pair::first_type >::argument_type [inherited]
argument_type is the type of the argument
```

result_type

```
typedef _Pair::first_type std::unary_function< _Pair , _Pair::first_type >::result_type [inherited]
result_type is the return type
```

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.919 __gnu_cxx::select2nd< _Pair > Struct Template Reference

Inherits `std::_Select2nd< _Pair >`.

Public Types

- typedef `_Pair` [argument_type](#)
- typedef `_Pair::second_type` [result_type](#)

Public Member Functions

- `_Pair::second_type & operator() (_Pair &__x) const`
- `const _Pair::second_type & operator() (const _Pair &__x) const`

6.919.1 Detailed Description

```
template<class _Pair>
struct __gnu_cxx::select2nd< _Pair >
```

An [SGI extension](#) .

6.919.2 Member Typedef Documentation

argument_type

```
typedef _Pair std::unary_function< _Pair , _Pair::second_type >::argument_type [inherited]
argument_type is the type of the argument
```

result_type

```
typedef _Pair::second_type std::unary_function< _Pair , _Pair::second_type >::result_type [inherited]
result_type is the return type
```

The documentation for this struct was generated from the following file:

- [ext/functional](#)

6.920 `__gnu_pbds::detail::select_value_type< Key, Mapped >` Struct Template Reference

```
#include <types_traits.hpp>
```

Public Types

- typedef `std::pair< const Key, Mapped >` **type**

6.920.1 Detailed Description

```
template<typename Key, typename Mapped>
struct __gnu_pbds::detail::select_value_type< Key, Mapped >
```

Choose `value_type` to be a key/value pair or just a key.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.921 `__gnu_pbds::detail::select_value_type< Key, null_type >` Struct Template Reference

```
#include <types_traits.hpp>
```

Public Types

- typedef `Key` **type**

6.921.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::select_value_type< Key, null_type >
```

Specialization for sets where the key is the `value_type`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.922 `std::basic_istream< _CharT, _Traits >::sentry` Class Reference**Public Types**

- typedef `__istream_type::__ctype_type` **__ctype_type**
- typedef `_Traits::int_type` **__int_type**
- typedef `basic_istream< _CharT, _Traits >` **__istream_type**
- typedef `basic_streambuf< _CharT, _Traits >` **__streambuf_type**
- typedef `_Traits traits_type`

Public Member Functions

- [sentry](#) (`basic_istream< _CharT, _Traits > &__is`, `bool __noskipws=false`)
- [operator bool](#) () const

6.922.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_istream< _CharT, _Traits >::sentry
```

Performs setup work for input streams.

Objects of this class are created before all of the standard extractors are run. It is responsible for *exception-safe prefix and suffix operations*, although only prefix actions are currently required by the standard.

6.922.2 Member Typedef Documentation

traits_type

```
template<typename _CharT , typename _Traits >
typedef _Traits std::basic\_istream< _CharT, _Traits >::sentry::traits_type
```

Easy access to dependent types.

6.922.3 Constructor & Destructor Documentation

sentry()

```
template<typename _CharT , typename _Traits >
std::basic\_istream< _CharT, _Traits >::sentry::sentry (
    basic\_istream< _CharT, _Traits > & __is,
    bool __noskipws = false ) [explicit]
```

The constructor performs all the work.

Parameters

<code>__is</code>	The input stream to guard.
<code>__noskipws</code>	Whether to consume whitespace or not.

If the stream state is good (`__is.good()` is true), then the following actions are performed, otherwise the sentry state is false (*not okay*) and failbit is set in the stream state.

The sentry's preparatory actions are:

1. if the stream is tied to an output stream, `is.tie()->flush()` is called to synchronize the output sequence
2. if `__noskipws` is false, and `ios_base::skipws` is set in `is.flags()`, the sentry extracts and discards whitespace characters from the stream. The currently imbued locale is used to determine whether each character is whitespace.

If the stream state is still good, then the sentry state becomes true (*okay*).

References [std::ios_base::badbit](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::flags\(\)](#), [std::basic_ios<_CharT, _Traits>::goodbit](#), [std::__ctype_abstract_base<_CharT>::is\(\)](#), [std::basic_ios<_CharT, _Traits>::rdbuf\(\)](#), [std::basic_ios<_CharT, _Traits>::setstate\(\)](#), [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#), [std::ios_base::skipws](#), [std::basic_streambuf<_CharT, _Traits>::snextc\(\)](#), and [std::basic_ios<_CharT, _Traits>::tie\(\)](#).

6.922.4 Member Function Documentation

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic\_istream< _CharT, _Traits >::sentry::operator bool ( ) const [inline], [explicit]
```

Quick status checking.

Returns

The sentry state.

For ease of use, sentries may be converted to booleans. The return value is that of the sentry state (true == okay). The documentation for this class was generated from the following files:

- [istream](#)
- [istream.tcc](#)

6.923 std::basic_ostream< _CharT, _Traits >::sentry Class Reference

Public Member Functions

- [sentry](#) ([basic_ostream](#)< _CharT, _Traits > &__os)
- [~sentry](#) ()
- [operator bool](#) () const

6.923.1 Detailed Description

template<typename _CharT, typename _Traits>
class std::basic_ostream< _CharT, _Traits >::sentry

Performs setup work for output streams.

Objects of this class are created before all of the standard inserters are run. It is responsible for *exception-safe prefix and suffix operations*.

6.923.2 Constructor & Destructor Documentation

sentry()

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits >::sentry::sentry (
    basic_ostream< _CharT, _Traits > & __os ) [explicit]
```

The constructor performs preparatory work.

Parameters

<code>__os</code>	The output stream to guard.
-------------------	-----------------------------

If the stream state is good (`__os.good()` is true), then if the stream is tied to another output stream, `is.↵tie() -> flush()` is called to synchronize the output sequences.

If the stream state is still good, then the sentry state becomes true (*okay*).

References [std::basic_ios< _CharT, _Traits >::bad\(\)](#), [std::ios_base::failbit](#), [std::basic_ios< _CharT, _Traits >::good\(\)](#), [std::basic_ios< _CharT, _Traits >::setstate\(\)](#), and [std::basic_ios< _CharT, _Traits >::tie\(\)](#).

~sentry()

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits >::sentry::~sentry ( ) [inline]
```

Possibly flushes the stream.

If `ios_base::unitbuf` is set in `os.flags()`, and `std::uncaught_exception()` is true, the sentry destructor calls `flush()` on the output stream.

6.923.3 Member Function Documentation

operator bool()

```
template<typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits >::sentry::operator bool ( ) const [inline], [explicit]
```

Quick status checking.

Returns

The sentry state.

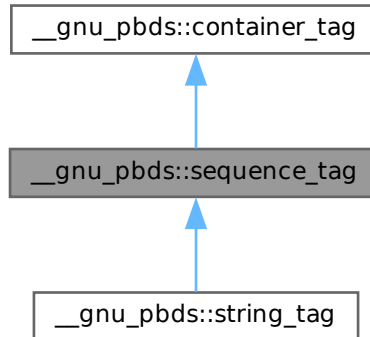
For ease of use, sentries may be converted to booleans. The return value is that of the sentry state (`true == okay`). The documentation for this class was generated from the following files:

- [ostream](#)
- [ostream.tcc](#)

6.924 `__gnu_pbds::sequence_tag` Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::sequence_tag`:



6.924.1 Detailed Description

Basic sequence.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.925 `__gnu_parallel::sequential_tag` Struct Reference

```
#include <tags.h>
```

6.925.1 Detailed Description

Forces sequential execution at compile time.

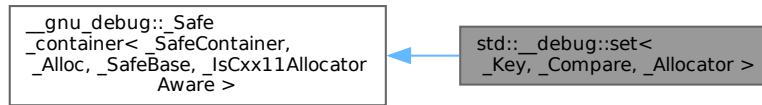
The documentation for this struct was generated from the following file:

- [tags.h](#)

6.926 `std::__debug::set<_Key, _Compare, _Allocator>` Class Template Reference

```
#include <set.h>
```

Inheritance diagram for std::__debug::set< _Key, _Compare, _Allocator >:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator< _Base_const_iterator, set >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- using `insert_return_type` = `_Node_insert_return< iterator, node_type >`
- typedef `__gnu_debug::_Safe_iterator< _Base_iterator, set >` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- using `node_type` = `typename _Base::node_type`
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Compare` **value_compare**
- typedef `_Key` **value_type**

Public Member Functions

- **set** (`_Base_ref __x`)
- `template<typename InputIterator >`
set (`_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator()`)
- `template<typename InputIterator >`
set (`_InputIterator __first, _InputIterator __last, const allocator_type &__a`)
- **set** (`const _Compare &__comp, const _Allocator &__a=_Allocator()`)
- **set** (`const allocator_type &__a`)
- **set** (`const set &`)=default
- **set** (`const set &__x, const __type_identity_t< allocator_type > &__a`)
- **set** (`(initializer_list< value_type > __l, const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type()`)
- **set** (`(initializer_list< value_type > __l, const allocator_type &__a`)
- **set** (`(set &&)=default`
- **set** (`(set &&__x, const __type_identity_t< allocator_type > &__a) noexcept(noexcept(_Base(std::move(__x), __a)))`)
- `const _Base & _M_base () const` `noexcept`
- `_Base & _M_base ()` `noexcept`

- `const_iterator` **begin** () const noexcept
- `iterator` **begin** () noexcept
- `const_iterator` **cbegin** () const noexcept
- `const_iterator` **cend** () const noexcept
- void **clear** () noexcept
- `const_reverse_iterator` **crbegin** () const noexcept
- `const_reverse_iterator` **crend** () const noexcept
- template<typename... _Args>
`std::pair< iterator, bool >` **emplace** (_Args &&... __args)
- template<typename... _Args>
`iterator` **emplace_hint** (const_iterator __pos, _Args &&... __args)
- `const_iterator` **end** () const noexcept
- `iterator` **end** () noexcept
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`std::pair< iterator, iterator >` **equal_range** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`std::pair< const_iterator, const_iterator >` **equal_range** (const _Kt &__x) const
- `std::pair< iterator, iterator >` **equal_range** (const key_type &__x)
- `std::pair< const_iterator, const_iterator >` **equal_range** (const key_type &__x) const
- `_Base_iterator` **erase** (_Base_const_iterator __position)
- size_type **erase** (const key_type &__x)
- _GLIBCXX_ABI_TAG_CXX11 `iterator` **erase** (const_iterator __first, const_iterator __last)
- _GLIBCXX_ABI_TAG_CXX11 `iterator` **erase** (const_iterator __position)
- node_type **extract** (const key_type &__key)
- node_type **extract** (const_iterator __position)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`iterator` **find** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`const_iterator` **find** (const _Kt &__x) const
- `iterator` **find** (const key_type &__x)
- `const_iterator` **find** (const key_type &__x) const
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- `std::pair< iterator, bool >` **insert** (const value_type &__x)
- `iterator` **insert** (const_iterator __hint, node_type &&__nh)
- `iterator` **insert** (const_iterator __position, const value_type &__x)
- `iterator` **insert** (const_iterator __position, value_type &&__x)
- void **insert** (initializer_list< value_type > __l)
- `insert_return_type` **insert** (node_type &&__nh)
- `std::pair< iterator, bool >` **insert** (value_type &&__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`iterator` **lower_bound** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
`const_iterator` **lower_bound** (const _Kt &__x) const
- `iterator` **lower_bound** (const key_type &__x)
- `const_iterator` **lower_bound** (const key_type &__x) const
- `set & operator=` (const `set` &)=default
- `set & operator=` (initializer_list< value_type > __l)
- `set & operator=` (`set` &&)=default
- `const_reverse_iterator` **rbegin** () const noexcept
- `reverse_iterator` **rbegin** () noexcept

- [const_reverse_iterator rend](#) () const noexcept
- [reverse_iterator rend](#) () noexcept
- void [swap](#) ([set](#) &__x) noexcept(*/*conditional */*)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[iterator upper_bound](#) (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
[const_iterator upper_bound](#) (const _Kt &__x) const
- [iterator upper_bound](#) (const key_type &__x)
- [const_iterator upper_bound](#) (const key_type &__x) const

Protected Member Functions

- void [_M_swap](#) (_Safe_container &__x) noexcept

Friends

- template<typename _ItT, typename _SeqT, typename _CatT>
class ::[__gnu_debug::Safe_iterator](#)

6.926.1 Detailed Description

template<typename _Key, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<[_↵](#)
_Key>>

class std::[__debug::set](#)<_Key, _Compare, _Allocator>

Class std::set with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [set.h](#)

6.927 std::set<_Key, _Compare, _Alloc> Class Template Reference

```
#include <stl_set.h>
```

Public Types

- using [insert_return_type](#) = typename [_Rep_type::insert_return_type](#)
- using [node_type](#) = typename [_Rep_type::node_type](#)
- typedef [_Key](#) [key_type](#)
- typedef [_Key](#) [value_type](#)
- typedef [_Compare](#) [key_compare](#)
- typedef [_Compare](#) [value_compare](#)
- typedef [_Alloc](#) [allocator_type](#)
- typedef [_Alloc_traits::pointer](#) [pointer](#)
- typedef [_Alloc_traits::const_pointer](#) [const_pointer](#)
- typedef [_Alloc_traits::reference](#) [reference](#)
- typedef [_Alloc_traits::const_reference](#) [const_reference](#)
- typedef [_Rep_type::const_iterator](#) [iterator](#)
- typedef [_Rep_type::const_iterator](#) [const_iterator](#)
- typedef [_Rep_type::const_reverse_iterator](#) [reverse_iterator](#)
- typedef [_Rep_type::const_reverse_iterator](#) [const_reverse_iterator](#)
- typedef [_Rep_type::size_type](#) [size_type](#)
- typedef [_Rep_type::difference_type](#) [difference_type](#)

Public Member Functions

- [set](#) ()=default
- [template<typename _InputIterator >](#)
[set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [template<typename _InputIterator >](#)
[set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp, const [allocator_type](#) &__a←
[a=allocator_type](#)())
- [template<typename _InputIterator >](#)
[set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- [set](#) (const [_Compare](#) &__comp, const [allocator_type](#) &__a=allocator_type())
- [set](#) (const [allocator_type](#) &__a)
- [set](#) (const [set](#) &)=default
- [set](#) (const [set](#) &__x, const [__type_identity_t](#)< [allocator_type](#) > &__a)
- [set](#) ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__comp=_Compare(), const [allocator_type](#) &__a←
[a=allocator_type](#)())
- [set](#) ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- [set](#) ([set](#) &&)=default
- [set](#) ([set](#) &&__x, const [__type_identity_t](#)< [allocator_type](#) > &__a) noexcept(is_nothrow_copy_constructible< [_Compare](#) >::value && [_Alloc_traits::S_always_equal](#)())
- [~set](#) ()=default
- [iterator begin](#) () const noexcept
- [iterator cbegin](#) () const noexcept
- [iterator cend](#) () const noexcept
- void [clear](#) () noexcept
- [reverse_iterator crbegin](#) () const noexcept
- [reverse_iterator crend](#) () const noexcept
- [template<typename... _Args>](#)
[std::pair](#)< [iterator](#), bool > [emplace](#) ([_Args](#) &&... __args)
- [template<typename... _Args>](#)
[iterator](#) [emplace_hint](#) (const [iterator](#) __pos, [_Args](#) &&... __args)
- bool [empty](#) () const noexcept
- [iterator end](#) () const noexcept
- [size_type](#) [erase](#) (const [key_type](#) &__x)
- [_GLIBCXX_ABI_TAG_CXX11](#) [iterator](#) [erase](#) (const [iterator](#) __first, const [iterator](#) __last)
- [_GLIBCXX_ABI_TAG_CXX11](#) [iterator](#) [erase](#) (const [iterator](#) __position)
- [node_type](#) [extract](#) (const [key_type](#) &__x)
- [node_type](#) [extract](#) (const [iterator](#) __pos)
- [allocator_type](#) [get_allocator](#) () const noexcept
- [template<typename _InputIterator >](#)
void [insert](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [std::pair](#)< [iterator](#), bool > [insert](#) (const [value_type](#) &__x)
- [iterator](#) [insert](#) (const [iterator](#) __hint, [node_type](#) &&__nh)
- [iterator](#) [insert](#) (const [iterator](#) __position, const [value_type](#) &__x)
- [iterator](#) [insert](#) (const [iterator](#) __position, [value_type](#) &&__x)
- void [insert](#) ([initializer_list](#)< [value_type](#) > __l)
- [insert_return_type](#) [insert](#) ([node_type](#) &&__nh)
- [std::pair](#)< [iterator](#), bool > [insert](#) ([value_type](#) &&__x)
- [key_compare](#) [key_comp](#) () const
- [size_type](#) [max_size](#) () const noexcept
- [template<typename _Compare1 >](#)
void [merge](#) ([multiset](#)< [_Key](#), [_Compare1](#), [_Alloc](#) > &&__source)

- `template<typename _Compare1 >`
`void merge (multiset< _Key, _Compare1, _Alloc > &__source)`
- `template<typename _Compare1 >`
`void merge (set< _Key, _Compare1, _Alloc > &&__source)`
- `template<typename _Compare1 >`
`void merge (set< _Key, _Compare1, _Alloc > &__source)`
- `set & operator= (const set &)=default`
- `set & operator= (initializer_list< value_type > __l)`
- `set & operator= (set &&)=default`
- `reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () const noexcept`
- `size_type size () const noexcept`
- `void swap (set &__x) noexcept(/*conditional */)`
- `value_compare value_comp () const`

- `size_type count (const key_type &__x) const`
- `template<typename _Kt >`
`auto count (const _Kt &__x) const -> decltype(_M_t._M_count_tr(__x))`

- `bool contains (const key_type &__x) const`
- `template<typename _Kt >`
`auto contains (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x), void(), true)`

- `iterator find (const key_type &__x)`
- `const_iterator find (const key_type &__x) const`
- `template<typename _Kt >`
`auto find (const _Kt &__x) -> decltype(iterator{ _M_t._M_find_tr(__x)})`
- `template<typename _Kt >`
`auto find (const _Kt &__x) const -> decltype(const_iterator{ _M_t._M_find_tr(__x)})`

- `iterator lower_bound (const key_type &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `template<typename _Kt >`
`auto lower_bound (const _Kt &__x) -> decltype(iterator{ _M_t._M_lower_bound_tr(__x)})`
- `template<typename _Kt >`
`auto lower_bound (const _Kt &__x) const -> decltype(const_iterator{ _M_t._M_lower_bound_tr(__x)})`

- `iterator upper_bound (const key_type &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt >`
`auto upper_bound (const _Kt &__x) -> decltype(iterator{ _M_t._M_upper_bound_tr(__x)})`
- `template<typename _Kt >`
`auto upper_bound (const _Kt &__x) const -> decltype(iterator{ _M_t._M_upper_bound_tr(__x)})`

- `std::pair< iterator, iterator > equal_range (const key_type &__x)`
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__x) const`
- `template<typename _Kt >`
`auto equal_range (const _Kt &__x) -> decltype(pair< iterator, iterator >{ _M_t._M_equal_range_tr(__x)})`
- `template<typename _Kt >`
`auto equal_range (const _Kt &__x) const -> decltype(pair< iterator, iterator >{ _M_t._M_equal_range_tr(__x)})`

Friends

- `template<typename _K1, typename _C1, typename _A1 >`
`bool operator< (const set< _K1, _C1, _A1 > &, const set< _K1, _C1, _A1 > &)`
- `template<typename _K1, typename _C1, typename _A1 >`
`bool operator== (const set< _K1, _C1, _A1 > &, const set< _K1, _C1, _A1 > &)`
- `template<typename, typename >`
`struct std::_Rb_tree_merge_helper`

6.927.1 Detailed Description

`template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>`
`class std::set< _Key, _Compare, _Alloc >`

A standard container made up of unique keys, which can be retrieved in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using unique keys).

Sets support bidirectional iterators.

The private tree data is declared exactly the same way for set and multiset; the distinction is made entirely in how the tree functions are called (*_unique versus *_equal, same as the standard).

6.927.2 Member Typedef Documentation

`allocator_type`

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
typedef _Alloc std::set< _Key, _Compare, _Alloc >::allocator_type
```

Public typedefs.

`const_iterator`

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
typedef _Rep_type::const_iterator std::set< _Key, _Compare, _Alloc >::const_iterator
```

Iterator-related typedefs.

`const_pointer`

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
typedef _Alloc_traits::const_pointer std::set< _Key, _Compare, _Alloc >::const_pointer
```

Iterator-related typedefs.

`const_reference`

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
typedef _Alloc_traits::const_reference std::set< _Key, _Compare, _Alloc >::const_reference
```

Iterator-related typedefs.

const_reverse_iterator

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Rep_type::const_reverse_iterator std::set< _Key, _Compare, _Alloc >::const_reverse_iterator
```

Iterator-related typedefs.

difference_type

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Rep_type::difference_type std::set< _Key, _Compare, _Alloc >::difference_type
```

Iterator-related typedefs.

iterator

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Rep_type::const_iterator std::set< _Key, _Compare, _Alloc >::iterator
```

Iterator-related typedefs.

key_compare

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Compare std::set< _Key, _Compare, _Alloc >::key_compare
```

Public typedefs.

key_type

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Key std::set< _Key, _Compare, _Alloc >::key_type
```

Public typedefs.

pointer

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Alloc_traits::pointer std::set< _Key, _Compare, _Alloc >::pointer
```

Iterator-related typedefs.

reference

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Alloc_traits::reference std::set< _Key, _Compare, _Alloc >::reference
```

Iterator-related typedefs.

reverse_iterator

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Rep_type::const_reverse_iterator std::set< _Key, _Compare, _Alloc >::reverse_iterator
Iterator-related typedefs.
```

size_type

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Rep_type::size_type std::set< _Key, _Compare, _Alloc >::size_type
Iterator-related typedefs.
```

value_compare

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Compare std::set< _Key, _Compare, _Alloc >::value_compare
Public typedefs.
```

value_type

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
typedef _Key std::set< _Key, _Compare, _Alloc >::value_type
Public typedefs.
```

6.927.3 Constructor & Destructor Documentation**set()** [1/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set ( ) [default]
Default constructor creates no elements.
```

set() [2/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
Creates a set with no elements.
```

Parameters

<code>__comp</code>	Comparator to use.
<code>__a</code>	An allocator object.

set() [3/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
template<typename _InputIterator >
std::set< _Key, _Compare, _Alloc >::set (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

Builds a set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a set consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

set() [4/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator >
std::set< _Key, _Compare, _Alloc >::set (
    _InputIterator __first,
    _InputIterator __last,
    const _Compare & __comp,
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a set consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

set() [5/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    const set< _Key, _Compare, _Alloc > & ) [default]
```

Set copy constructor.

Whether the allocator is copied depends on the allocator traits.

set() [6/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    set< _Key, _Compare, _Alloc > && ) [default]
```

Set move constructor

The newly-created set contains the exact contents of the moved instance. The moved instance is a valid, but unspecified, set.

set() [7/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    initializer_list< value_type > __l,
    const _Compare & __comp = _Compare(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds a set from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a set consisting of copies of the elements in the list. This is linear in N if the list is already sorted, and NlogN otherwise (where N is `__l.size()`).

set() [8/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    const allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

set() [9/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    const set< _Key, _Compare, _Alloc > & __x,
    const __type_identity_t< allocator_type > & __a ) [inline]
```

Allocator-extended copy constructor.

set() [10/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    set< _Key, _Compare, _Alloc > && __x,
    const __type_identity_t< allocator_type > & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

set() [11/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (
    initializer_list< value_type > __l,
    const allocator_type & __a ) [inline]
```

Allocator-extended initializer-list constructor.

set() [12/12]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator >
std::set<_Key, _Compare, _Alloc>::set (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a ) [inline]
```

Allocator-extended range constructor.

~set()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set<_Key, _Compare, _Alloc>::~~set ( ) [default]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

6.927.4 Member Function Documentation

begin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc>::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the set. Iteration is done in ascending order according to the keys.

cbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc>::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the set. Iteration is done in ascending order according to the keys.

cend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc>::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the set. Iteration is done in ascending order according to the keys.

clear()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
void std::set<_Key, _Compare, _Alloc>::clear ( ) [inline], [noexcept]
```

Erases all elements in a set. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x), void()), true)    [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_Kt</code> <code>__x</code>	Key of elements to be located.
--------------------------------------	--------------------------------

Returns

True if there is an element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
bool std::set< _Key, _Compare, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_Kt</code> <code>__x</code>	Key of elements to be located.
--------------------------------------	--------------------------------

Returns

True if there is an element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::count (
    const _Kt & __x ) const -> decltype(_M_t._M_count_tr(__x))    [inline]
```

Finds the number of elements.

Parameters

<code>_Kt</code> <code>__x</code>	Element to located.
--------------------------------------	---------------------

Returns

Number of elements with specified key.

This function only makes sense for multisets; for set the result will either be 0 (not present) or 1 (present).

count() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::set<_Key, _Compare, _Alloc>::count (
    const key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

<code>__x</code>	Element to located.
------------------	---------------------

Returns

Number of elements with specified key.

This function only makes sense for multisets; for set the result will either be 0 (not present) or 1 (present).

crbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::set<_Key, _Compare, _Alloc>::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the last element in the set. Iteration is done in descending order according to the keys.

crend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::set<_Key, _Compare, _Alloc>::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the set. Iteration is done in descending order according to the keys.

emplace()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename... _Args>
std::pair< iterator, bool > std::set<_Key, _Compare, _Alloc>::emplace (
    _Args &&... __args ) [inline]
```

Attempts to build and insert an element into the set.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to build and insert an element into the set. A set relies on unique keys and thus an element is only inserted if it is not already present in the set.

Insertion requires logarithmic time.

emplace_hint()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename... _Args>
iterator std::set< _Key, _Compare, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Attempts to insert an element into the set.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the element with key equivalent to the one generated from `__args` (may or may not be the element itself).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

empty()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
bool std::set< _Key, _Compare, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the set is empty.

end()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set< _Key, _Compare, _Alloc >::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the set. Iteration is done in ascending order according to the keys.

Referenced by `std::set< _Key, _Compare, _Alloc >::extract()`.

equal_range() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(pair<iterator, iterator>)(_M_t._M_equal_range_tr(__x))
[inline]
```

Finds a subsequence matching given key.

Parameters

<code>_Key</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set<_Key, _Compare, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(pair<iterator, iterator>(_M_t._M_equal_range_
tr(__x)))    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_Key</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::pair< iterator, iterator > std::set<_Key, _Compare, _Alloc >::equal_range (
    const key_type & __x )    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_Key</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

equal_range() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
std::pair< const_iterator, const_iterator > std::set< _Key, _Compare, _Alloc >::equal_range (
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>↵</code>	Key to be located.
<code>__X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

erase() [1/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
size_type std::set< _Key, _Compare, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>↵</code>	Key of element to be erased.
<code>__X</code>	

Returns

The number of elements erased.

This function erases all the elements located by the given key from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
_GLIBCXX_ABI_TAG_CXX11 iterator std::set<_Key, _Compare, _Alloc>::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [`__first`,`__last`) range of elements from a set.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
_GLIBCXX_ABI_TAG_CXX11 iterator std::set<_Key, _Compare, _Alloc>::erase (
    const_iterator __position ) [inline]
```

Erases an element from a set.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
node_type std::set<_Key, _Compare, _Alloc>::extract (
    const key_type & __x ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
node_type std::set<_Key, _Compare, _Alloc>::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::set<_Key, _Compare, _Alloc>::end()`.

find() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::find (
    const _Kt & __x ) -> decltype(iterator{_M_t._M_find_tr(__x)})    [inline]
```

Tries to locate an element in a set.

Parameters

<code>↵</code> <code>__x</code>	Element to be located.
------------------------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::find (
    const _Kt & __x ) const -> decltype(const_iterator{_M_t._M_find_tr(__x)})    [inline]
```

Tries to locate an element in a set.

Parameters

<code>↵</code> <code>__x</code>	Element to be located.
------------------------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
iterator std::set< _Key, _Compare, _Alloc >::find (
    const key_type & __x )    [inline]
```

Tries to locate an element in a set.

Parameters

<code>↵</code> <code>__x</code>	Element to be located.
------------------------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
const_iterator std::set< _Key, _Compare, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in a set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

get_allocator()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
allocator_type std::set< _Key, _Compare, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Returns the allocator object with which the set was constructed.

insert() [1/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator >
void std::set< _Key, _Compare, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
std::pair< iterator, bool > std::set< _Key, _Compare, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Attempts to insert an element into the set.

Parameters

<code>__x</code>	Element to be inserted.
------------------	-------------------------

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to insert an element into the set. A set relies on unique keys and thus an element is only inserted if it is not already present in the set.

Insertion requires logarithmic time.

References `std::pair< _T1, _T2 >::first`, and `std::pair< _T1, _T2 >::second`.

Referenced by `std::set< _Key, _Compare, _Alloc >::insert()`.

insert() [3/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set< _Key, _Compare, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References `std::move()`.

insert() [4/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set< _Key, _Compare, _Alloc >::insert (
    const_iterator __position,
    const value_type & __x ) [inline]
```

Attempts to insert an element into the set.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of `__x` (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

insert() [5/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
void std::set<_Key, _Compare, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of elements into the set.

Parameters

↩	A std::initializer_list<value_type> of elements to be inserted.
↩	
↩	
↩	
/	

Complexity similar to that of the range constructor.

References [std::set<_Key, _Compare, _Alloc >::insert\(\)](#).

insert() [6/6]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
insert_return_type std::set<_Key, _Compare, _Alloc >::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

key_comp()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
key_compare std::set<_Key, _Compare, _Alloc >::key_comp ( ) const [inline]
```

Returns the comparison object with which the set was constructed.

lower_bound() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set<_Key, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) -> decltype(iterator(_M.t._M_lower_bound_tr(__x))) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

↩	Key to be located.
_X	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
template<typename _Kt >
auto std::set< _Key, _Compare, _Alloc >::lower_bound (
    const _Kt & __x ) const -> decltype(const_iterator(_M_t._M_lower_bound_tr(__x)))
[inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
iterator std::set< _Key, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

lower_bound() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
const_iterator std::set< _Key, _Compare, _Alloc >::lower_bound (
    const key_type & __x ) const [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

max_size()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
size_type std::set<_Key, _Compare, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the set.

operator=() [1/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
set & std::set<_Key, _Compare, _Alloc >::operator= (
    const set<_Key, _Compare, _Alloc > & ) [default]
```

Set assignment operator.

Whether the allocator is copied depends on the allocator traits.

operator=() [2/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
set & std::set<_Key, _Compare, _Alloc >::operator= (
    initializer_list<value_type> & __l ) [inline]
```

Set list assignment operator.

Parameters

↵	An initializer_list.
↵	
↵	
↵	
/	

This function fills a set with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the set and that the resulting set's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
set & std::set<_Key, _Compare, _Alloc >::operator= (
    set<_Key, _Compare, _Alloc > && ) [default]
```

Move assignment operator.

rbegin()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```



```
reverse_iterator std::set< _Key, _Compare, _Alloc >::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the last element in the set. Iteration is done in descending order according to the keys.

rend()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
reverse_iterator std::set< _Key, _Compare, _Alloc >::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the set. Iteration is done in descending order according to the keys.

size()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
size_type std::set< _Key, _Compare, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the size of the set.

swap()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
void std::set< _Key, _Compare, _Alloc >::swap (
    set< _Key, _Compare, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another set.

Parameters

<code>↵</code>	A set of the same element and allocator types.
<code>__x</code>	

This exchanges the elements between two sets in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

upper_bound() [1/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<↵
_Key>>
```

```
template<typename _Kt >
```

```
auto std::set< _Key, _Compare, _Alloc >::upper_bound (
    const _Kt & __x ) -> decltype(iterator(_M_t._M_upper_bound_tr(__x))) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>↵</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [2/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt >
auto std::set<_Key, _Compare, _Alloc>::upper_bound (
    const _Kt & __x ) const -> decltype(iterator(_M_t._M_upper_bound_tr(__x)))    [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [3/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc>::upper_bound (
    const key_type & __x ) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

upper_bound() [4/4]

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
const_iterator std::set<_Key, _Compare, _Alloc>::upper_bound (
    const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

value_comp()

```
template<typename _Key , typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
```

```
value_compare std::set< _Key, _Compare, _Alloc >::value_comp ( ) const [inline]
```

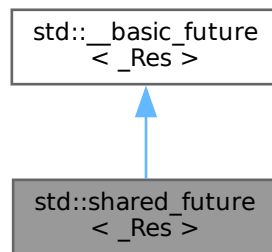
Returns the comparison object with which the set was constructed.

The documentation for this class was generated from the following files:

- [stl_multiset.h](#)
- [stl_set.h](#)

6.928 std::shared_future< _Res > Class Template Reference

Inheritance diagram for std::shared_future< _Res >:

**Public Member Functions**

- `shared_future` (const `shared_future` &__sf) noexcept
- `shared_future` (future< _Res > &&__uf) noexcept
- `shared_future` (shared_future &&__sf) noexcept
- const _Res & `get` () const
- `shared_future` & `operator=` (const `shared_future` &__sf) noexcept
- `shared_future` & `operator=` (shared_future &&__sf) noexcept
- bool `valid` () const noexcept
- void `wait` () const
- template<typename _Rep , typename _Period >
future_status `wait_for` (const chrono::duration< _Rep, _Period > &__rel) const
- template<typename _Clock , typename _Duration >
future_status `wait_until` (const chrono::time_point< _Clock, _Duration > &__abs) const

Protected Types

- typedef __future_base::Result< _Res > & __result_type
- typedef shared_ptr< _State_base > __state_type

Protected Member Functions

- `__result_type M_get_result () const`
- `void M_swap (__basic_future & __that) noexcept`

6.928.1 Detailed Description

```
template<typename _Res>
class std::shared_future<_Res>
```

Primary template for shared_future.

6.928.2 Constructor & Destructor Documentation**shared_future() [1/3]**

```
template<typename _Res>
std::shared_future<_Res>::shared_future (
    const shared_future<_Res> & __sf ) [inline], [noexcept]
```

Copy constructor.

shared_future() [2/3]

```
template<typename _Res>
std::shared_future<_Res>::shared_future (
    future<_Res> && __uf ) [inline], [noexcept]
```

Construct from a future rvalue.

shared_future() [3/3]

```
template<typename _Res>
std::shared_future<_Res>::shared_future (
    shared_future<_Res> && __sf ) [inline], [noexcept]
```

Construct from a shared_future rvalue.

6.928.3 Member Function Documentation**M_get_result()**

```
template<typename _Res>
__result_type std::__basic_future<_Res>::M_get_result ( ) const [inline], [protected], [inherited]
```

Wait for the state to be ready and rethrow any stored exception.

get()

```
template<typename _Res>
const _Res & std::shared_future<_Res>::get ( ) const [inline]
```

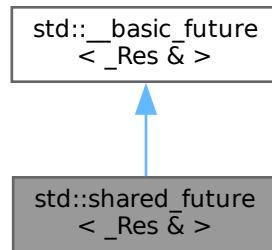
Retrieving the value.

The documentation for this class was generated from the following file:

- [future](#)

6.929 std::shared_future<_Res & > Class Template Reference

Inheritance diagram for std::shared_future<_Res & >:



Public Member Functions

- `shared_future` (const `shared_future` &__sf)
- `shared_future` (`future`<_Res & > &&__uf) noexcept
- `shared_future` (`shared_future` &&__sf) noexcept
- `_Res` & `get` () const
- `shared_future` & `operator=` (const `shared_future` &__sf)
- `shared_future` & `operator=` (`shared_future` &&__sf) noexcept
- bool `valid` () const noexcept
- void `wait` () const
- `future_status` `wait_for` (const `chrono::duration`<_Rep, _Period > &__rel) const
- `future_status` `wait_until` (const `chrono::time_point`<_Clock, _Duration > &__abs) const

Protected Types

- typedef `__future_base::_Result`<_Res & > & `__result_type`
- typedef `shared_ptr`<_State_base > `__state_type`

Protected Member Functions

- `__result_type` `_M_get_result` () const
- void `_M_swap` (`__basic_future` &__that) noexcept

6.929.1 Detailed Description

```
template<typename _Res>
class std::shared_future<_Res & >
```

Partial specialization for `shared_future`<R&>

6.929.2 Constructor & Destructor Documentation

shared_future() [1/3]

```
template<typename _Res >
std::shared_future<_Res & >::shared_future (
    const shared_future<_Res & > & __sf ) [inline]
```

Copy constructor.

shared_future() [2/3]

```
template<typename _Res >
std::shared_future<_Res & >::shared_future (
    future<_Res & > && __uf ) [inline], [noexcept]
```

Construct from a future rvalue.

shared_future() [3/3]

```
template<typename _Res >
std::shared_future<_Res & >::shared_future (
    shared_future<_Res & > && __sf ) [inline], [noexcept]
```

Construct from a shared_future rvalue.

6.929.3 Member Function Documentation

_M_get_result()

```
__result_type std::__basic_future<_Res & >::M_get_result ( ) const [inline], [protected],
[inherited]
```

Wait for the state to be ready and rethrow any stored exception.

get()

```
template<typename _Res >
_Res & std::shared_future<_Res & >::get ( ) const [inline]
```

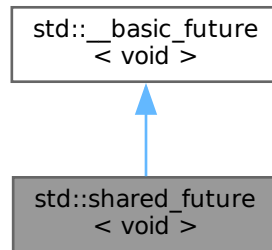
Retrieving the value.

The documentation for this class was generated from the following file:

- [future](#)

6.930 std::shared_future< void > Class Reference

Inheritance diagram for std::shared_future< void >:



Public Member Functions

- [shared_future](#) (const [shared_future](#) &__sf)
- [shared_future](#) (future< void > &&__uf) noexcept
- [shared_future](#) ([shared_future](#) &&__sf) noexcept
- void **get** () const
- [shared_future](#) & **operator=** (const [shared_future](#) &__sf)
- [shared_future](#) & **operator=** ([shared_future](#) &&__sf) noexcept
- bool **valid** () const noexcept
- void **wait** () const
- [future_status](#) **wait_for** (const [chrono::duration](#)< _Rep, _Period > &__rel) const
- [future_status](#) **wait_until** (const [chrono::time_point](#)< _Clock, _Duration > &__abs) const

Protected Types

- typedef __future_base::_Result< void > & __**result_type**
- typedef [shared_ptr](#)< _State_base > __**state_type**

Protected Member Functions

- __result_type **_M_get_result** () const
- void **_M_swap** (__[basic_future](#) &__that) noexcept

6.930.1 Detailed Description

Explicit specialization for shared_future<void>

6.930.2 Constructor & Destructor Documentation

shared_future() [1/3]

```
std::shared_future< void >::shared_future (
    const shared\_future< void > & __sf ) [inline]
```

Copy constructor.

shared_future() [2/3]

```
std::shared_future< void >::shared_future (
    future< void > && __uf ) [inline], [noexcept]
```

Construct from a future rvalue.

shared_future() [3/3]

```
std::shared_future< void >::shared_future (
    shared_future< void > && __sf ) [inline], [noexcept]
```

Construct from a shared_future rvalue.

6.930.3 Member Function Documentation**__M_get_result()**

```
__result_type std::__basic_future< void >::__M_get_result ( ) const [inline], [protected], [inherited]
```

Wait for the state to be ready and rethrow any stored exception.

The documentation for this class was generated from the following file:

- [future](#)

6.931 std::shared_lock< _Mutex > Class Template Reference**Public Types**

- typedef `_Mutex` `mutex_type`

Public Member Functions

- **shared_lock** (`mutex_type` &__m)
- **shared_lock** (`mutex_type` &__m, [adopt_lock_t](#))
- template<typename `_Rep`, typename `_Period` >
shared_lock (`mutex_type` &__m, const [chrono::duration](#)< `_Rep`, `_Period` > &__rel_time)
- template<typename `_Clock`, typename `_Duration` >
shared_lock (`mutex_type` &__m, const [chrono::time_point](#)< `_Clock`, `_Duration` > &__abs_time)
- **shared_lock** (`mutex_type` &__m, [defer_lock_t](#)) noexcept
- **shared_lock** (`mutex_type` &__m, [try_to_lock_t](#))
- **shared_lock** ([shared_lock](#) &&__sl) noexcept
- **shared_lock** ([shared_lock](#) const &)=delete
- void **lock** ()
- `mutex_type` * **mutex** () const noexcept
- **operator bool** () const noexcept
- [shared_lock](#) & **operator=** ([shared_lock](#) &&__sl) noexcept
- [shared_lock](#) & **operator=** ([shared_lock](#) const &)=delete
- bool **owns_lock** () const noexcept
- `mutex_type` * **release** () noexcept
- void **swap** ([shared_lock](#) &__u) noexcept
- bool **try_lock** ()
- template<typename `_Rep`, typename `_Period` >
bool **try_lock_for** (const [chrono::duration](#)< `_Rep`, `_Period` > &__rel_time)
- template<typename `_Clock`, typename `_Duration` >
bool **try_lock_until** (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__abs_time)
- void **unlock** ()

6.931.1 Detailed Description

```
template<typename _Mutex>
class std::shared_lock< _Mutex >
```

shared_lock

The documentation for this class was generated from the following file:

- [shared_mutex](#)

6.932 std::shared_mutex Class Reference

Public Member Functions

- **shared_mutex** (const [shared_mutex](#) &)=delete
- void **lock** ()
- void **lock_shared** ()
- [shared_mutex](#) & **operator=** (const [shared_mutex](#) &)=delete
- bool **try_lock** ()
- bool **try_lock_shared** ()
- void **unlock** ()
- void **unlock_shared** ()

Related Symbols

(Note that these are not member symbols.)

- template<typename _Mutex >
void [swap](#) ([shared_lock](#)< _Mutex > &__x, [shared_lock](#)< _Mutex > &__y) noexcept

6.932.1 Detailed Description

The standard shared mutex type.

The documentation for this class was generated from the following file:

- [shared_mutex](#)

6.933 std::shared_ptr< _Tp > Class Template Reference

```
#include <memory>
```

Inherits std::__shared_ptr< _Tp, _Lp >.

Public Types

- using [element_type](#) = typename __shared_ptr< _Tp >::element_type
- using [weak_type](#) = [weak_ptr](#)< _Tp >

Public Member Functions

- constexpr [shared_ptr](#) () noexcept
- template<typename _Yp, typename = _Constructible<_Yp*>>
[shared_ptr](#) (_Yp *__p)
- template<typename _Yp, typename _Deleter, typename = _Constructible<_Yp*, _Deleter>>
[shared_ptr](#) (_Yp *__p, _Deleter __d)
- template<typename _Yp, typename _Deleter, typename _Alloc, typename = _Constructible<_Yp*, _Deleter, _Alloc>>
[shared_ptr](#) (_Yp *__p, _Deleter __d, _Alloc __a)

- template<typename _Yp, typename = _Constructible<auto_ptr<_Yp>>>>
 shared_ptr (auto_ptr<_Yp> &&__r)
- **shared_ptr** (const **shared_ptr** &) noexcept=default
- template<typename _Yp, typename = _Constructible<const shared_ptr<_Yp>&>>>
 shared_ptr (const **shared_ptr**<_Yp> &__r) noexcept
- template<typename _Yp>
 shared_ptr (const **shared_ptr**<_Yp> &__r, element_type *__p) noexcept
- template<typename _Yp, typename = _Constructible<const weak_ptr<_Yp>&>>>
 shared_ptr (const **weak_ptr**<_Yp> &__r)
- template<typename _Deleter>
 shared_ptr (nullptr_t __p, _Deleter __d)
- template<typename _Deleter, typename _Alloc>
 shared_ptr (nullptr_t __p, _Deleter __d, _Alloc __a)
- constexpr **shared_ptr** (nullptr_t) noexcept
- **shared_ptr** (**shared_ptr** &&__r) noexcept
- template<typename _Yp, typename = _Constructible<shared_ptr<_Yp>>>>
 shared_ptr (**shared_ptr**<_Yp> &&__r) noexcept
- template<typename _Yp>
 shared_ptr (**shared_ptr**<_Yp> &&__r, element_type *__p) noexcept
- template<typename _Tp1, typename>
 shared_ptr (std::auto_ptr<_Tp1> &&__r)
- template<typename _Yp, typename _Del, typename = _Constructible<unique_ptr<_Yp, _Del>>>>
 shared_ptr (**unique_ptr**<_Yp, _Del> &&__r)
- element_type * **get** () const noexcept
- operator bool () const noexcept
- element_type & operator* () const noexcept
- element_type * operator-> () const noexcept
- template<typename _Yp>
 _Assignable< auto_ptr<_Yp> > operator= (auto_ptr<_Yp> &&__r)
- **shared_ptr** & operator= (const **shared_ptr** &) noexcept=default
- template<typename _Yp>
 _Assignable< const **shared_ptr**<_Yp> > operator= (const **shared_ptr**<_Yp> &__r) noexcept
- **shared_ptr** & operator= (**shared_ptr** &&__r) noexcept
- template<class _Yp>
 _Assignable< **shared_ptr**<_Yp> > operator= (**shared_ptr**<_Yp> &&__r) noexcept
- template<typename _Yp, typename _Del>
 _Assignable< **unique_ptr**<_Yp, _Del> > operator= (**unique_ptr**<_Yp, _Del> &&__r)
- void **reset** () noexcept
- template<typename _Yp>
 _SafeConv<_Yp> **reset** (_Yp *__p)
- template<typename _Yp, typename _Deleter>
 _SafeConv<_Yp> **reset** (_Yp *__p, _Deleter __d)
- template<typename _Yp, typename _Deleter, typename _Alloc>
 _SafeConv<_Yp> **reset** (_Yp *__p, _Deleter __d, _Alloc __a)
- void **swap** (__shared_ptr<_Tp, _Lp> &__other) noexcept
- bool **unique** () const noexcept
- long **use_count** () const noexcept
- template<typename _Tp1>
 bool **owner_before** (__shared_ptr<_Tp1, _Lp> const &__rhs) const noexcept
- template<typename _Tp1>
 bool **owner_before** (__weak_ptr<_Tp1, _Lp> const &__rhs) const noexcept

Friends

- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_BoundedArray<_Yp>> allocate_shared (const _Alloc &)`
- `template<typename _Yp, typename _Alloc, typename... _Args>`
`shared_ptr<_NonArray<_Yp>> allocate_shared (const _Alloc &, _Args &&...)`
- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_BoundedArray<_Yp>> allocate_shared (const _Alloc &, const remove_extent_t<_Yp> &)`
- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_UnboundedArray<_Yp>> allocate_shared (const _Alloc &, size_t)`
- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_UnboundedArray<_Yp>> allocate_shared (const _Alloc &, size_t, const remove_extent_t<_Yp> &)`
- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_NotUnboundedArray<_Yp>> allocate_shared_for_overwrite (const _Alloc &)`
- `template<typename _Yp, typename _Alloc >`
`shared_ptr<_UnboundedArray<_Yp>> allocate_shared_for_overwrite (const _Alloc &, size_t)`
- `template<typename _Yp >`
`shared_ptr<_BoundedArray<_Yp>> make_shared ()`
- `template<typename _Yp, typename... _Args>`
`shared_ptr<_NonArray<_Yp>> make_shared (_Args &&...)`
- `template<typename _Yp >`
`shared_ptr<_BoundedArray<_Yp>> make_shared (const remove_extent_t<_Yp> &)`
- `template<typename _Yp >`
`shared_ptr<_UnboundedArray<_Yp>> make_shared (size_t)`
- `template<typename _Yp >`
`shared_ptr<_UnboundedArray<_Yp>> make_shared (size_t, const remove_extent_t<_Yp> &)`
- `template<typename _Yp >`
`shared_ptr<_NotUnboundedArray<_Yp>> make_shared_for_overwrite ()`
- `template<typename _Yp >`
`shared_ptr<_UnboundedArray<_Yp>> make_shared_for_overwrite (size_t)`
- `class weak_ptr<_Tp >`

Related Symbols

(Note that these are not member symbols.)

- `template<typename _Del, typename _Tp >`
`_Del * get_deleter (const shared_ptr<_Tp> &__p) noexcept`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream<_Ch, _Tr> & operator<< (std::basic_ostream<_Ch, _Tr> &__os, const __shared_ptr<_Tp, _Lp> &__p)`
- `template<typename _Tp, typename _Up >`
`bool operator== (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept`
- `template<typename _Tp >`
`bool operator== (const shared_ptr<_Tp> &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator== (nullptr_t, const shared_ptr<_Tp> &__a) noexcept`
- `template<typename _Tp, typename _Up >`
`bool operator!= (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept`
- `template<typename _Tp >`
`bool operator!= (const shared_ptr<_Tp> &__a, nullptr_t) noexcept`

- template<typename _Tp>
bool operator!= (nullptr_t, const shared_ptr<_Tp> &__a) noexcept
- template<typename _Tp, typename _Up>
bool operator< (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept
- template<typename _Tp>
bool operator< (const shared_ptr<_Tp> &__a, nullptr_t) noexcept
- template<typename _Tp>
bool operator< (nullptr_t, const shared_ptr<_Tp> &__a) noexcept
- template<typename _Tp, typename _Up>
bool operator<= (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept
- template<typename _Tp>
bool operator<= (const shared_ptr<_Tp> &__a, nullptr_t) noexcept
- template<typename _Tp>
bool operator<= (nullptr_t, const shared_ptr<_Tp> &__a) noexcept
- template<typename _Tp, typename _Up>
bool operator> (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept
- template<typename _Tp>
bool operator> (const shared_ptr<_Tp> &__a, nullptr_t) noexcept
- template<typename _Tp>
bool operator> (nullptr_t, const shared_ptr<_Tp> &__a) noexcept
- template<typename _Tp, typename _Up>
bool operator>= (const shared_ptr<_Tp> &__a, const shared_ptr<_Up> &__b) noexcept
- template<typename _Tp>
bool operator>= (const shared_ptr<_Tp> &__a, nullptr_t) noexcept
- template<typename _Tp>
bool operator>= (nullptr_t, const shared_ptr<_Tp> &__a) noexcept
- template<typename _Tp>
void swap (shared_ptr<_Tp> &__a, shared_ptr<_Tp> &__b) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> static_pointer_cast (const shared_ptr<_Up> &__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> const_pointer_cast (const shared_ptr<_Up> &__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> dynamic_pointer_cast (const shared_ptr<_Up> &__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> reinterpret_pointer_cast (const shared_ptr<_Up> &__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> static_pointer_cast (shared_ptr<_Up> &&__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> const_pointer_cast (shared_ptr<_Up> &&__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> dynamic_pointer_cast (shared_ptr<_Up> &&__r) noexcept
- template<typename _Tp, typename _Up>
shared_ptr<_Tp> reinterpret_pointer_cast (shared_ptr<_Up> &&__r) noexcept

- template<typename _Tp, typename _Alloc, typename... _Args>
shared_ptr<_NonArray<_Tp>> allocate_shared (const _Alloc &__a, _Args &&... __args)
- template<typename _Tp, typename... _Args>
shared_ptr<_NonArray<_Tp>> make_shared (_Args &&... __args)
- template<typename _Tp, typename _Alloc>
shared_ptr<_UnboundedArray<_Tp>> allocate_shared (const _Alloc &__a, size_t __n)

- `template<typename _Tp, typename _Alloc >`
`shared_ptr<_BoundedArray<_Tp > > allocate_shared (const _Alloc &__a)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_is_lock_free (const __shared_ptr<_Tp, _Lp > *__p)`
- `template<typename _Tp >`
`shared_ptr<_Tp > atomic_load_explicit (const shared_ptr<_Tp > *__p, memory_order)`
- `template<typename _Tp >`
`void atomic_store_explicit (shared_ptr<_Tp > *__p, shared_ptr<_Tp > __r, memory_order)`
- `template<typename _Tp >`
`shared_ptr<_Tp > atomic_exchange_explicit (shared_ptr<_Tp > *__p, shared_ptr<_Tp > __r, memory_order)`
- `template<typename _Tp >`
`bool atomic_compare_exchange_strong_explicit (shared_ptr<_Tp > *__p, shared_ptr<_Tp > *__v, shared_ptr<_Tp > __w, memory_order, memory_order)`

6.933.1 Detailed Description

`template<typename _Tp>`
class `std::shared_ptr<_Tp >`

A smart pointer with reference-counted copy semantics.

Since

C++11

A `shared_ptr` object is either empty or *owns* a pointer passed to the constructor. Copies of a `shared_ptr` share ownership of the same pointer. When the last `shared_ptr` that owns the pointer is destroyed or reset, the owned pointer is freed (either by `delete` or by invoking a custom deleter that was passed to the constructor).

A `shared_ptr` also stores another pointer, which is usually (but not always) the same pointer as it owns. The stored pointer can be retrieved by calling the `get()` member function.

The equality and relational operators for `shared_ptr` only compare the stored pointer returned by `get()`, not the owned pointer. To test whether two `shared_ptr` objects share ownership of the same pointer see `std::shared_ptr::owner_before` and `std::owner_less`.

6.933.2 Member Typedef Documentation

element_type

`template<typename _Tp >`
`using std::shared_ptr<_Tp >::element_type = typename __shared_ptr<_Tp>::element_type`
 The type pointed to by the stored pointer, `remove_extent_t<_Tp>`

weak_type

```
template<typename _Tp >
using std::shared_ptr< _Tp >::weak_type = weak_ptr<_Tp>
```

The corresponding weak_ptr type for this shared_ptr.

Since

C++17

6.933.3 Constructor & Destructor Documentation**shared_ptr() [1/14]**

```
template<typename _Tp >
constexpr std::shared_ptr< _Tp >::shared_ptr ( ) [inline], [constexpr], [noexcept]
```

Construct an empty shared_ptr.

Postcondition

use_count()==0 && get()==0

shared_ptr() [2/14]

```
template<typename _Tp >
std::shared_ptr< _Tp >::shared_ptr (
    const shared_ptr< _Tp > & ) [default], [noexcept]
```


Copy constructor.

shared_ptr() [3/14]

```
template<typename _Tp >
template<typename _Yp , typename = _Constructible<_Yp*>>
std::shared_ptr< _Tp >::shared_ptr (
    _Yp * __p ) [inline], [explicit]
```

Construct a shared_ptr that owns the pointer __p.

Parameters

 __p	A pointer that is convertible to element_type*.
--	---

Postcondition

use_count() == 1 && get() == __p

Exceptions

<i>std::bad_alloc</i> , in	which case delete __p is called.
----------------------------	----------------------------------

shared_ptr() [4/14]

```
template<typename _Tp >
template<typename _Yp , typename _Deleter , typename = _Constructible<_Yp*, _Deleter>>
```

```
std::shared_ptr< _Tp >::shared_ptr (
    _Yp * __p,
    _Deleter __d ) [inline]
```

Construct a `shared_ptr` that owns the pointer `__p` and the deleter `__d`.

Parameters

<code>__p</code>	A pointer.
<code>__d</code>	A deleter.

Postcondition

`use_count() == 1 && get() == __p`

Exceptions

<code>std::bad_alloc</code> , in	which case <code>__d(__p)</code> is called.
----------------------------------	---

Requirements: `_Deleter`'s copy constructor and destructor must not throw
`__shared_ptr` will release `__p` by calling `__d(__p)`

shared_ptr() [5/14]

```
template<typename _Tp >
template<typename _Deleter >
std::shared_ptr< _Tp >::shared_ptr (
    nullptr_t __p,
    _Deleter __d ) [inline]
```

Construct a `shared_ptr` that owns a null pointer and the deleter `__d`.

Parameters

<code>__p</code>	A null pointer constant.
<code>__d</code>	A deleter.

Postcondition

`use_count() == 1 && get() == __p`

Exceptions

<code>std::bad_alloc</code> , in	which case <code>__d(__p)</code> is called.
----------------------------------	---

Requirements: `_Deleter`'s copy constructor and destructor must not throw
The last owner will call `__d(__p)`

shared_ptr() [6/14]

```
template<typename _Tp >
template<typename _Yp , typename _Deleter , typename _Alloc , typename = _Constructible<_Yp*, _↵
Deleter, _Alloc>>
std::shared_ptr< _Tp >::shared_ptr (
    _Yp * __p,
    _Deleter __d,
    _Alloc __a ) [inline]
```

Construct a shared_ptr that owns the pointer __p and the deleter __d.

Parameters

<u>↵</u> __p	A pointer.
<u>↵</u> __d	A deleter.
<u>↵</u> __a	An allocator.

Postcondition

use_count() == 1 && get() == __p

Exceptions

<i>std::bad_alloc</i> , in	which case __d(__p) is called.
----------------------------	--------------------------------

Requirements: _Deleter's copy constructor and destructor must not throw _Alloc's copy constructor and destructor must not throw.

__shared_ptr will release __p by calling __d(__p)

shared_ptr() [7/14]

```
template<typename _Tp >
template<typename _Deleter , typename _Alloc >
std::shared_ptr< _Tp >::shared_ptr (
    nullptr_t __p,
    _Deleter __d,
    _Alloc __a ) [inline]
```

Construct a shared_ptr that owns a null pointer and the deleter __d.

Parameters

<u>↵</u> __p	A null pointer constant.
<u>↵</u> __d	A deleter.
<u>↵</u> __a	An allocator.

Postcondition

```
use_count() == 1 && get() == __p
```

Exceptions

<code>std::bad_alloc</code> , in	which case <code>__d(__p)</code> is called.
----------------------------------	---

Requirements: `_Deleter`'s copy constructor and destructor must not throw `_Alloc`'s copy constructor and destructor must not throw.

The last owner will call `__d(__p)`

shared_ptr() [8/14]

```
template<typename _Tp >
template<typename _Yp >
std::shared_ptr< _Tp >::shared_ptr (
    const shared_ptr< _Yp > & __r,
    element_type * __p ) [inline], [noexcept]
```

Constructs a `shared_ptr` instance that stores `__p` and shares ownership with `__r`.

Parameters

<code>__r</code>	A <code>shared_ptr</code> .
<code>__p</code>	A pointer that will remain valid while <code>*__r</code> is valid.

Postcondition

```
get() == __p && use_count() == __r.use_count()
```

This can be used to construct a `shared_ptr` to a sub-object of an object managed by an existing `shared_ptr`. The complete object will remain valid while any `shared_ptr` owns it, even if they don't store a pointer to the complete object.

```
shared_ptr<pair<int,int>> pii(new pair<int,int>());
shared_ptr<int> pi(pii, &pii->first);
assert(pii.use_count() == 2);
```

shared_ptr() [9/14]

```
template<typename _Tp >
template<typename _Yp >
std::shared_ptr< _Tp >::shared_ptr (
    shared_ptr< _Yp > && __r,
    element_type * __p ) [inline], [noexcept]
```

Constructs a `shared_ptr` instance that stores `__p` and shares ownership with `__r`.

Parameters

<code>__r</code>	A <code>shared_ptr</code> .
<code>__p</code>	A pointer that will remain valid while <code>*__r</code> is valid.

Postcondition

```
get() == __p && !__r.use_count() && !__r.get()
```

Since

C++17

This can be used to construct a `shared_ptr` to a sub-object of an object managed by an existing `shared_ptr`. The complete object will remain valid while any `shared_ptr` owns it, even if they don't store a pointer to the complete object.

```
shared_ptr<pair<int,int>> pii(new pair<int,int>());
shared_ptr<int> pi1(pii, &pii->first);
assert(pii.use_count() == 2);
shared_ptr<int> pi2(std::move(pii), &pii->second);
assert(pii.use_count() == 0);
```

shared_ptr() [10/14]

```
template<typename _Tp >
template<typename _Yp , typename = _Constructible<const shared_ptr<_Yp>&>>
std::shared_ptr< _Tp >::shared_ptr (
    const shared_ptr< _Yp > & __r ) [inline], [noexcept]
```

If `__r` is empty, constructs an empty `shared_ptr`; otherwise construct a `shared_ptr` that shares ownership with `__r`.

Parameters

↩	A shared_ptr.
↩	
↩	
↩	
<i>r</i>	

Postcondition

```
get() == __r.get() && use_count() == __r.use_count()
```

shared_ptr() [11/14]

```
template<typename _Tp >
std::shared_ptr< _Tp >::shared_ptr (
    shared_ptr< _Tp > && __r ) [inline], [noexcept]
```

Move-constructs a `shared_ptr` instance from `__r`.

Parameters

↩	A shared_ptr rvalue.
↩	
↩	
↩	
<i>r</i>	

Postcondition

*this contains the old value of `__r`, `__r` is empty.

shared_ptr() [12/14]

```
template<typename _Tp >
template<typename _Yp , typename = _Constructible<shared_ptr<_Yp>>>
std::shared_ptr< _Tp >::shared_ptr (
    shared_ptr< _Yp > && __r ) [inline], [noexcept]
```

Move-constructs a shared_ptr instance from __r.

Parameters

↵	A shared_ptr rvalue.
__↵	
↵	
__↵	
<i>r</i>	

Postcondition

*this contains the old value of __r, __r is empty.

shared_ptr() [13/14]

```
template<typename _Tp >
template<typename _Yp , typename = _Constructible<const weak_ptr<_Yp>>>
std::shared_ptr< _Tp >::shared_ptr (
    const weak_ptr< _Yp > & __r ) [inline], [explicit]
```

Constructs a shared_ptr that shares ownership with __r and stores a copy of the pointer stored in __r.

Parameters

↵	A weak_ptr.
__↵	
↵	
__↵	
<i>r</i>	

Postcondition

use_count() == __r.use_count()

Exceptions

<i>bad_weak_ptr</i>	when __r.expired(), in which case the constructor has no effect.
---------------------	--

shared_ptr() [14/14]

```
template<typename _Tp >
constexpr std::shared_ptr< _Tp >::shared_ptr (
    nullptr_t ) [inline], [constexpr], [noexcept]
```

Construct an empty shared_ptr.

Postcondition

```
use_count() == 0 && get() == nullptr
```

6.933.4 Member Function Documentation**get()**

```
template<typename _Tp , _Lock_policy _Lp>
element_type * std::__shared_ptr< _Tp, _Lp >::get ( ) const [inline], [noexcept], [inherited]
```

Return the stored pointer.

operator bool()

```
template<typename _Tp , _Lock_policy _Lp>
std::__shared_ptr< _Tp, _Lp >::operator bool ( ) const [inline], [explicit], [noexcept], [inherited]
```

Return true if the stored pointer is not null.

owner_before() [1/2]

```
template<typename _Tp , _Lock_policy _Lp>
template<typename _Tp1 >
bool std::__shared_ptr< _Tp, _Lp >::owner_before (
    __shared_ptr< _Tp1, _Lp > const & __rhs ) const [inline], [noexcept], [inherited]
```

Define an ordering based on ownership.

This function defines a strict weak ordering between two shared_ptr or weak_ptr objects, such that one object is less than the other unless they share ownership of the same pointer, or are both empty.

owner_before() [2/2]

```
template<typename _Tp , _Lock_policy _Lp>
template<typename _Tp1 >
bool std::__shared_ptr< _Tp, _Lp >::owner_before (
    __weak_ptr< _Tp1, _Lp > const & __rhs ) const [inline], [noexcept], [inherited]
```

Define an ordering based on ownership.

This function defines a strict weak ordering between two shared_ptr or weak_ptr objects, such that one object is less than the other unless they share ownership of the same pointer, or are both empty.

swap()

```
template<typename _Tp , _Lock_policy _Lp>
void std::__shared_ptr< _Tp, _Lp >::swap (
    __shared_ptr< _Tp, _Lp > & __other ) [inline], [noexcept], [inherited]
```

Exchange both the owned pointer and the stored pointer.

unique()

```
template<typename _Tp , _Lock_policy _Lp>
bool std::__shared_ptr< _Tp, _Lp >::unique ( ) const [inline], [noexcept], [inherited]
```

Return true if use_count() == 1.

use_count()

```
template<typename _Tp , _Lock_policy _Lp>
long std::__shared_ptr< _Tp, _Lp >::use_count ( ) const [inline], [noexcept], [inherited]
```

If *this owns a pointer, return the number of owners, otherwise zero.

The documentation for this class was generated from the following files:

- [bits/shared_ptr.h](#)
- [shared_ptr_atomic.h](#)
- [auto_ptr.h](#)

6.934 std::shared_timed_mutex Class Reference

Inherits `__shared_timed_mutex_base`.

Public Member Functions

- **shared_timed_mutex** (const [shared_timed_mutex](#) &)=delete
- void **lock** ()
- void **lock_shared** ()
- **shared_timed_mutex** & **operator=** (const [shared_timed_mutex](#) &)=delete
- bool **try_lock** ()
- template<typename _Rep, typename _Period >
bool **try_lock_for** (const [chrono::duration](#)< _Rep, _Period > &__rtime)
- bool **try_lock_shared** ()
- template<typename _Rep, typename _Period >
bool **try_lock_shared_for** (const [chrono::duration](#)< _Rep, _Period > &__rtime)
- template<typename _Clock, typename _Duration >
bool **try_lock_shared_until** (const [chrono::time_point](#)< _Clock, _Duration > &__abs_time)
- template<typename _Clock, typename _Duration >
bool **try_lock_until** (const [chrono::time_point](#)< _Clock, _Duration > &__abs_time)
- void **unlock** ()
- void **unlock_shared** ()

6.934.1 Detailed Description

The standard shared timed mutex type.

The documentation for this class was generated from the following file:

- [shared_mutex](#)

6.935 std::shuffle_order_engine< _RandomNumberEngine, __k > Class Template Reference

```
#include <random.h>
```

Public Types

- template<typename _Sseq >
using **_If_seed_seq** = typename [enable_if](#)< __detail::__is_seed_seq< _Sseq, [shuffle_order_engine](#), [result_type](#) >::value >::type
- typedef `_RandomNumberEngine::result_type` [result_type](#)

Public Member Functions

- [shuffle_order_engine](#) ()
- [shuffle_order_engine](#) (_RandomNumberEngine &&__rng)
- template<typename _Sseq, typename = _If_seed_seq<_Sseq>>
[shuffle_order_engine](#) (_Sseq &__q)
- [shuffle_order_engine](#) (const _RandomNumberEngine &__rng)
- [shuffle_order_engine](#) ([result_type](#) __s)

- const _RandomNumberEngine & [base](#) () const noexcept
- void [discard](#) (unsigned long long __z)
- [result_type](#) operator() ()
- void [seed](#) ()
- template<typename _Sseq >
_If_seed_seq<_Sseq > [seed](#) (_Sseq &__q)
- void [seed](#) ([result_type](#) __s)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr size_t [table_size](#)

Friends

- template<typename _RandomNumberEngine1, size_t __k1, typename _CharT, typename _Traits >
[std::basic_ostream](#)<_CharT, _Traits > & [operator<<](#) ([std::basic_ostream](#)<_CharT, _Traits > &__os, const
[std::shuffle_order_engine](#)<_RandomNumberEngine1, __k1 > &__x)
- bool [operator==](#) (const [shuffle_order_engine](#) &__lhs, const [shuffle_order_engine](#) &__rhs)
- template<typename _RandomNumberEngine1, size_t __k1, typename _CharT, typename _Traits >
[std::basic_istream](#)<_CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)<_CharT, _Traits > &__is,
[std::shuffle_order_engine](#)<_RandomNumberEngine1, __k1 > &__x)

6.935.1 Detailed Description

template<typename _RandomNumberEngine, size_t __k>
class std::shuffle_order_engine<_RandomNumberEngine, __k>

Produces random numbers by reordering random numbers from some base engine.

The values from the base engine are stored in a sequence of size __k and shuffled by an algorithm that depends on those values.

6.935.2 Member Typedef Documentation

result_type

```
template<typename _RandomNumberEngine, size_t __k>
typedef _RandomNumberEngine::result_type std::shuffle\_order\_engine<_RandomNumberEngine, __k>↵
::result_type
```

The type of the generated random value.

6.935.3 Constructor & Destructor Documentation

shuffle_order_engine() [1/5]

```
template<typename _RandomNumberEngine, size_t __k>
std::shuffle\_order\_engine<_RandomNumberEngine, __k>::shuffle_order_engine () [inline]
```

Constructs a default shuffle_order_engine engine.

The underlying engine is default constructed as well.

shuffle_order_engine() [2/5]

```
template<typename _RandomNumberEngine , size_t __k>
std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (
    const _RandomNumberEngine & __rng ) [inline], [explicit]
```

Copy constructs a shuffle_order_engine engine.
Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

shuffle_order_engine() [3/5]

```
template<typename _RandomNumberEngine , size_t __k>
std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (
    _RandomNumberEngine && __rng ) [inline], [explicit]
```

Move constructs a shuffle_order_engine engine.
Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

shuffle_order_engine() [4/5]

```
template<typename _RandomNumberEngine , size_t __k>
std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (
    result_type __s ) [inline], [explicit]
```

Seed constructs a shuffle_order_engine engine.
Constructs the underlying generator engine seeded with __s.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

shuffle_order_engine() [5/5]

```
template<typename _RandomNumberEngine , size_t __k>
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (
    _Sseq & __q ) [inline], [explicit]
```

Generator construct a shuffle_order_engine engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

6.935.4 Member Function Documentation

base()

```
template<typename _RandomNumberEngine , size_t __k>
const _RandomNumberEngine & std::shuffle_order_engine< _RandomNumberEngine, __k >::base ( ) const
[inline], [noexcept]
```

Gets a const reference to the underlying generator engine object.

discard()

```
template<typename _RandomNumberEngine , size_t __k>
void std::shuffle_order_engine< _RandomNumberEngine, __k >::discard (
    unsigned long long __z ) [inline]
```

Discard a sequence of random numbers.

max()

```
template<typename _RandomNumberEngine , size_t __k>
static constexpr result_type std::shuffle_order_engine< _RandomNumberEngine, __k >::max ( ) [inline],
[static], [constexpr]
```

Gets the maximum value in the generated random number range.

min()

```
template<typename _RandomNumberEngine , size_t __k>
static constexpr result_type std::shuffle_order_engine< _RandomNumberEngine, __k >::min ( ) [inline],
[static], [constexpr]
```

Gets the minimum value in the generated random number range.

operator>()

```
template<typename _RandomNumberEngine , size_t __k>
shuffle_order_engine< _RandomNumberEngine, __k >::result_type std::shuffle_order_engine< _↵
_RandomNumberEngine, __k >::operator()
```

Gets the next value in the generated random number sequence.

References [std::max\(\)](#), and [std::min\(\)](#).

seed() [1/3]

```
template<typename _RandomNumberEngine , size_t __k>
void std::shuffle_order_engine< _RandomNumberEngine, __k >::seed ( ) [inline]
```

Reseeds the shuffle_order_engine object with the default seed for the underlying base class generator engine.

seed() [2/3]

```
template<typename _RandomNumberEngine , size_t __k>
template<typename _Sseq >
_If_seed_seq< _Sseq > std::shuffle_order_engine< _RandomNumberEngine, __k >::seed (
    _Sseq & __q ) [inline]
```

Reseeds the shuffle_order_engine object with the given seed sequence.

Parameters

_↵	A seed generator function.
_q	

seed() [3/3]

```
template<typename _RandomNumberEngine , size_t __k>
void std::shuffle_order_engine< _RandomNumberEngine, __k >::seed (
    result_type __s ) [inline]
```

Reseeds the shuffle_order_engine object with the default seed for the underlying base class generator engine.

6.935.5 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RandomNumberEngine , size_t __k>
template<typename _RandomNumberEngine1 , size_t __k1, typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::shuffle_order_engine< _RandomNumberEngine1, __k1 > & __x ) [friend]
```

Inserts the current state of a shuffle_order_engine random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A shuffle_order_engine random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RandomNumberEngine , size_t __k>
bool operator== (
    const shuffle_order_engine< _RandomNumberEngine, __k > & __lhs,
    const shuffle_order_engine< _RandomNumberEngine, __k > & __rhs ) [friend]
```

Compares two shuffle_order_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A shuffle_order_engine random number generator object.
<code>__rhs</code>	Another shuffle_order_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

operator>>

```
template<typename _RandomNumberEngine , size_t __k>
template<typename _RandomNumberEngine1 , size_t __k1, typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::shuffle_order_engine< _RandomNumberEngine1, __k1 > & __x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine `__x` from the input stream `__is`.

Parameters

<code>_is</code>	An input stream.
<code>_x</code>	A shuffle_order_engine random number generator engine.

Returns

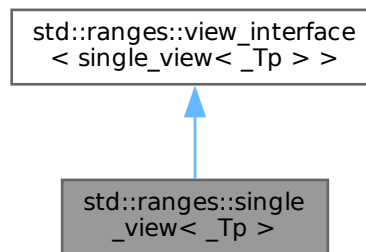
The input stream with the state of `_x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.936 std::ranges::single_view< _Tp > Class Template Reference

Inheritance diagram for std::ranges::single_view< _Tp >:



Public Member Functions

- constexpr **single_view** (_Tp &&__t) noexcept(is_nothrow_move_constructible_v< _Tp >)
- constexpr **single_view** (const _Tp &&__t) noexcept(is_nothrow_copy_constructible_v< _Tp >)
- template<typename... _Args>
requires constructible_from< _Tp, _Args... >
constexpr **single_view** (in_place_t, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, _Args... >)
- constexpr decltype(auto) **back** ()
- constexpr decltype(auto) **back** () const
- constexpr const _Tp * **begin** () const noexcept
- constexpr _Tp * **begin** () noexcept
- constexpr const _Tp * **data** () const noexcept
- constexpr _Tp * **data** () noexcept
- constexpr bool **empty** () const noexcept(noexcept(_S_empty(_M_derived())))
- constexpr bool **empty** () noexcept(noexcept(_S_empty(_M_derived())))
- constexpr const _Tp * **end** () const noexcept
- constexpr _Tp * **end** () noexcept

- constexpr decltype(auto) **front** ()
- constexpr decltype(auto) **front** () const
- constexpr **operator bool** () const noexcept(noexcept(ranges::empty(_M_derived())))
- constexpr **operator bool** () noexcept(noexcept(ranges::empty(_M_derived())))
- constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n)
- constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n) const
- constexpr auto **size** () const noexcept(noexcept(_S_size(_M_derived())))

Static Public Member Functions

- static constexpr size_t **size** () noexcept

6.936.1 Detailed Description

```
template<copy_constructible _Tp>
requires is_object_v<_Tp>
class std::ranges::single_view< _Tp >
```

A view that contains exactly one element.

The documentation for this class was generated from the following file:

- [ranges](#)

6.937 std::slice Class Reference

```
#include <slice_array.h>
```

Public Member Functions

- [slice](#) ()
- [slice](#) (size_t __o, size_t __d, size_t __s)
- size_t [size](#) () const
- size_t [start](#) () const
- size_t [stride](#) () const

Friends

- bool [operator==](#) (const [slice](#) &, const [slice](#) &)=default

6.937.1 Detailed Description

Class defining one-dimensional subset of an array.

The slice class represents a one-dimensional subset of an array, specified by three parameters: start offset, size, and stride. The start offset is the index of the first element of the array that is part of the subset. The size is the total number of elements in the subset. Stride is the distance between each successive array element to include in the subset.

For example, with an array of size 10, and a slice with offset 1, size 3 and stride 2, the subset consists of array elements 1, 3, and 5.

6.937.2 Friends And Related Symbol Documentation

operator==

```
bool operator== (
    const slice & ,
    const slice & ) [friend]
```

Equality comparison.

The documentation for this class was generated from the following file:

- [slice_array.h](#)

6.938 std::slice_array<_Tp> Class Template Reference

```
#include <slice_array.h>
```

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- [slice_array](#) (const [slice_array](#) &)
- template<class `_Dom` >
void **operator%=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator%=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator&=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator&=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator*=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator*=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator+=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator+=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator-=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator-=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator/=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator/=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator<<=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator<<=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator=](#) (const `_Tp` &) const
- [slice_array](#) & [operator=](#) (const [slice_array](#) &)
- void [operator=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator>>=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator>>=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator^=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator^=](#) (const [valarray](#)< `_Tp` > &) const
- template<class `_Dom` >
void **operator|=** (const `_Expr`< `_Dom`, `_Tp` > &) const
- void [operator|=](#) (const [valarray](#)< `_Tp` > &) const

Friends

- class [valarray](#)< `_Tp` >

6.938.1 Detailed Description

```
template<typename _Tp>
class std::slice_array<_Tp >
```

Reference to one-dimensional subset of an array.

A slice_array is a reference to the actual elements of an array specified by a slice. The way to get a slice_array is to call operator[](slice) on a valarray. The returned slice_array then permits carrying operations out on the referenced subset of elements in the original valarray. For example, operator+=(valarray) will add values to the subset of elements in the underlying valarray this slice_array refers to.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

6.938.2 Member Function Documentation

operator%=()

```
template<typename _Tp >
void std::slice_array<_Tp >::operator%= (
    const valarray<_Tp > & ) const
```

Modulo slice elements by corresponding elements of *v*.

operator&=()

```
template<typename _Tp >
void std::slice_array<_Tp >::operator&= (
    const valarray<_Tp > & ) const
```

Logical and slice elements with corresponding elements of *v*.

operator*=()

```
template<typename _Tp >
void std::slice_array<_Tp >::operator*= (
    const valarray<_Tp > & ) const
```

Multiply slice elements by corresponding elements of *v*.

operator+=()

```
template<typename _Tp >
void std::slice_array<_Tp >::operator+= (
    const valarray<_Tp > & ) const
```

Add corresponding elements of *v* to slice elements.

operator-=()

```
template<typename _Tp >
void std::slice_array<_Tp >::operator-= (
    const valarray<_Tp > & ) const
```

Subtract corresponding elements of *v* from slice elements.

operator/=()

```
template<typename _Tp >
void std::slice_array< _Tp >::operator/= (
    const valarray< _Tp > & ) const
```

Divide slice elements by corresponding elements of *v*.

operator<<=()

```
template<typename _Tp >
void std::slice_array< _Tp >::operator<<= (
    const valarray< _Tp > & ) const
```

Left shift slice elements by corresponding elements of *v*.

operator>>=()

```
template<typename _Tp >
void std::slice_array< _Tp >::operator>>= (
    const valarray< _Tp > & ) const
```

Right shift slice elements by corresponding elements of *v*.

operator^=()

```
template<typename _Tp >
void std::slice_array< _Tp >::operator^= (
    const valarray< _Tp > & ) const
```

Logical xor slice elements with corresponding elements of *v*.

operator" |= ()

```
template<typename _Tp >
void std::slice_array< _Tp >::operator|= (
    const valarray< _Tp > & ) const
```

Logical or slice elements with corresponding elements of *v*.

The documentation for this class was generated from the following files:

- [valarray](#)
- [slice_array.h](#)

6.939 `__gnu_cxx::slist< _Tp, _Alloc >` Class Template Reference

Inherits `__gnu_cxx::Slist_base< _Tp, _Alloc >`.

Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `_Slist_iterator< _Tp, const _Tp &, const _Tp * >` **const_iterator**
- typedef `const value_type *` **const_pointer**
- typedef `const value_type &` **const_reference**
- typedef `std::ptrdiff_t` **difference_type**
- typedef `_Slist_iterator< _Tp, _Tp &, _Tp * >` **iterator**
- typedef `value_type *` **pointer**
- typedef `value_type &` **reference**
- typedef `std::size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `template<class _InputIterator >`
`slist` (`_InputIterator __first`, `_InputIterator __last`, `const allocator_type &__a=allocator_type()`)
- `slist` (`const allocator_type &__a=allocator_type()`)
- `slist` (`const slist &__x`)
- `slist` (`size_type __n`)
- `slist` (`size_type __n`, `const value_type &__x`, `const allocator_type &__a=allocator_type()`)
- `template<class _InputIterator >`
`void _M_assign_dispatch` (`_InputIterator __first`, `_InputIterator __last`, `std::__false_type`)
- `template<class _Integer >`
`void _M_assign_dispatch` (`_Integer __n`, `_Integer __val`, `std::__true_type`)
- `void _M_fill_assign` (`size_type __n`, `const _Tp &__val`)
- `template<class _InputIterator >`
`void assign` (`_InputIterator __first`, `_InputIterator __last`)
- `void assign` (`size_type __n`, `const _Tp &__val`)
- `iterator before_begin` ()
- `const_iterator before_begin` () `const`
- `iterator begin` ()
- `const_iterator begin` () `const`
- `void clear` ()
- `bool empty` () `const`
- `iterator end` ()
- `const_iterator end` () `const`
- `iterator erase` (`iterator __first`, `iterator __last`)
- `iterator erase` (`iterator __pos`)
- `iterator erase_after` (`iterator __before_first`, `iterator __last`)
- `iterator erase_after` (`iterator __pos`)
- `reference front` ()
- `const_reference front` () `const`
- `allocator_type get_allocator` () `const`
- `iterator insert` (`iterator __pos`)
- `template<class _InIterator >`
`void insert` (`iterator __pos`, `_InIterator __first`, `_InIterator __last`)
- `iterator insert` (`iterator __pos`, `const value_type &__x`)
- `void insert` (`iterator __pos`, `size_type __n`, `const value_type &__x`)
- `iterator insert_after` (`iterator __pos`)
- `template<class _InIterator >`
`void insert_after` (`iterator __pos`, `_InIterator __first`, `_InIterator __last`)
- `iterator insert_after` (`iterator __pos`, `const value_type &__x`)
- `void insert_after` (`iterator __pos`, `size_type __n`, `const value_type &__x`)
- `size_type max_size` () `const`
- `template<class _StrictWeakOrdering >`
`void merge` (`slist &`, `_StrictWeakOrdering`)
- `void merge` (`slist &__x`)
- `slist & operator=` (`const slist &__x`)
- `void pop_front` ()
- `iterator previous` (`const_iterator __pos`)
- `const_iterator previous` (`const_iterator __pos`) `const`
- `void push_front` ()
- `void push_front` (`const value_type &__x`)
- `void remove` (`const _Tp &__val`)

- template<class _Predicate >
void **remove_if** (_Predicate __pred)
- void **resize** (size_type new_size)
- void **resize** (size_type new_size, const _Tp &__x)
- void **reverse** ()
- size_type **size** () const
- void **sort** ()
- template<class _StrictWeakOrdering >
void **sort** (_StrictWeakOrdering __comp)
- void **splice** (iterator __pos, [slist](#) &__x)
- void **splice** (iterator __pos, [slist](#) &__x, iterator __first, iterator __last)
- void **splice** (iterator __pos, [slist](#) &__x, iterator __i)
- void **splice_after** (iterator __pos, iterator __before_first, iterator __before_last)
- void **splice_after** (iterator __pos, iterator __prev)
- void **splice_after** (iterator __pos, [slist](#) &__x)
- void **swap** ([slist](#) &__x)
- void **unique** ()
- template<class _BinaryPredicate >
void **unique** (_BinaryPredicate __pred)

6.939.1 Detailed Description

```
template<class _Tp, class _Alloc = std::allocator<_Tp>>
class __gnu_cxx::slist< _Tp, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

The documentation for this class was generated from the following file:

- [slist](#)

6.940 std::source_location Struct Reference

Public Member Functions

- constexpr uint_least32_t **column** () const noexcept
- constexpr const char * **file_name** () const noexcept
- constexpr const char * **function_name** () const noexcept
- constexpr uint_least32_t **line** () const noexcept

Static Public Member Functions

- static constexpr [source_location](#) **current** (__builtin_ret_type __p=__builtin_source_location()) noexcept

6.940.1 Detailed Description

A class that describes a location in source code.

The documentation for this struct was generated from the following file:

- [source_location](#)

6.941 `std::experimental::filesystem::v1::space_info` Struct Reference

```
#include <fs_fwd.h>
```

Public Attributes

- `uintmax_t available`
- `uintmax_t capacity`
- `uintmax_t free`

6.941.1 Detailed Description

Information about free space on a disk.

The documentation for this struct was generated from the following file:

- [experimental/bits/fs_fwd.h](#)

6.942 `std::filesystem::space_info` Struct Reference

```
#include <fs_fwd.h>
```

Public Attributes

- `uintmax_t available`
- `uintmax_t capacity`
- `uintmax_t free`

Friends

- `bool operator==` (const [space_info](#) &, const [space_info](#) &)=default

6.942.1 Detailed Description

Information about free space on a disk.

The documentation for this struct was generated from the following file:

- [bits/fs_fwd.h](#)

6.943 `__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

```
#include <splay_tree_.hpp>
```

Inherits `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`.

Public Types

- `typedef _Alloc allocator_type`
- `typedef Cmp_Fn cmp_fn`
- `typedef std::pair< size_type, size_type > comp_hash`
- `typedef base_type::const_iterator const_iterator`
- `typedef base_type::const_pointer const_pointer`
- `typedef base_type::const_reference const_reference`
- `typedef base_type::const_reverse_iterator const_reverse_iterator`
- `typedef splay_tree_tag container_category`
- `typedef _Alloc::difference_type difference_type`

- `typedef base_type::iterator` **iterator**
- `typedef base_type::key_const_pointer` **key_const_pointer**
- `typedef base_type::key_const_reference` **key_const_reference**
- `typedef base_type::key_pointer` **key_pointer**
- `typedef base_type::key_reference` **key_reference**
- `typedef base_type::key_type` **key_type**
- `typedef base_type::mapped_const_pointer` **mapped_const_pointer**
- `typedef base_type::mapped_const_reference` **mapped_const_reference**
- `typedef base_type::mapped_pointer` **mapped_pointer**
- `typedef base_type::mapped_reference` **mapped_reference**
- `typedef base_type::mapped_type` **mapped_type**
- `typedef __nothrowcopy::indicator` **no_throw_indicator**
- `typedef traits_type::node_const_iterator` **node_const_iterator**
- `typedef traits_type::node_iterator` **node_iterator**
- `typedef base_type::node_update` **node_update**
- `typedef base_type::const_iterator` **point_const_iterator**
- `typedef base_type::point_iterator` **point_iterator**
- `typedef base_type::pointer` **pointer**
- `typedef base_type::reference` **reference**
- `typedef base_type::reverse_iterator` **reverse_iterator**
- `typedef _Alloc::size_type` **size_type**
- `typedef integral_constant< int, Store_Hash >` **store_extra**
- `typedef stored_data< value_type, size_type, Store_Hash >` **stored_data_type**
- `typedef base_type::value_type` **value_type**

Public Member Functions

- **splay_tree_map** (const Cmp_Fn &)
- **splay_tree_map** (const Cmp_Fn &, const node_update &)
- **splay_tree_map** (const [splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- iterator **erase** (iterator it)
- bool **erase** (key_const_reference)
- reverse_iterator **erase** (reverse_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- void **initialize** ()
- `std::pair< point_iterator, bool >` **insert** (const_reference r_value)
- void **join** ([splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- point_iterator **lower_bound** (key_const_reference)

- point_const_iterator **lower_bound** (key_const_reference) const
- size_type **max_size** () const
- node_iterator **node_begin** ()
- node_const_iterator **node_begin** () const
- node_iterator **node_end** ()
- node_const_iterator **node_end** () const
- mapped_reference **operator[]** (key_const_reference r_key)
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- size_type **size** () const
- void **split** (key_const_reference, [splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **swap** ([splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **swap** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference)
- point_const_iterator **upper_bound** (key_const_reference) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Types

- typedef node_alloc_traits::value_type **node**
- typedef node_alloc_traits::allocator_type **node_allocator**
- typedef traits_type::null_node_update_pointer **null_node_update_pointer**
- typedef [types_traits](#)< Key, Mapped, _Alloc, false > **traits_base**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- template<typename Node_Update_>
void **apply_update** (node_pointer, Node_Update_*)
- void **apply_update** (node_pointer, null_node_update_pointer)
- [std::pair](#)< node_pointer, bool > **erase** (node_pointer)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, false_type)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, true_type)
- void **initialize_min_max** ()
- iterator **insert_imp_empty** (const_reference)
- [std::pair](#)< point_iterator, bool > **insert_leaf** (const_reference)
- iterator **insert_leaf_new** (const_reference, node_pointer, bool)
- void **join_finish** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- bool **join_prep** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- size_type **recursive_count** (node_pointer) const
- void **rotate_left** (node_pointer)
- void **rotate_parent** (node_pointer)
- void **rotate_right** (node_pointer)
- void **split_finish** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)
- bool **split_prep** (key_const_reference, [tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)

- void **update_min_max_for_erased_node** (node_pointer)
- template<typename Node_Update_>
void **update_to_top** (node_pointer, Node_Update_ *)
- void **update_to_top** (node_pointer, null_node_update_pointer)
- void **value_swap** ([tree_order_statistics_node_update](#)< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > &)

Static Protected Member Functions

- static void **clear_imp** (node_pointer)

Protected Attributes

- node_pointer **m_p_head**
- size_type **m_size**

Static Protected Attributes

- static node_allocator **s_node_allocator**

6.943.1 Detailed Description

template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename [_↵](#)
_Alloc>

class `__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`

Splay tree.

6.943.2 Member Function Documentation

node_begin() [1/2]

template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >

node_iterator `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, ↵
_Alloc >::node_begin ()` [inline], [inherited]

Returns a node_iterator corresponding to the node at the root of the tree.

node_begin() [2/2]

template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >

node_const_iterator `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_↵
Traits, _Alloc >::node_begin () const` [inline], [inherited]

Returns a const node_iterator corresponding to the node at the root of the tree.

node_end() [1/2]

template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >

node_iterator `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, ↵
_Alloc >::node_end ()` [inline], [inherited]

Returns a node_iterator corresponding to a node just after a leaf of the tree.

node_end() [2/2]

```
template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc >
```

```
node_const_iterator __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end ( ) const [inline], [inherited]
```

Returns a const node_iterator corresponding to a node just after a leaf of the tree.

The documentation for this class was generated from the following file:

- [splay_tree.hpp](#)

6.944 __gnu_pbds::detail::splay_tree_node_< Value_Type, Metadata, _Alloc > Struct Template Reference

```
#include <node.hpp>
```

Public Types

- typedef [rebind_traits](#)< _Alloc, metadata_type >::const_reference **metadata_const_reference**
- typedef [rebind_traits](#)< _Alloc, metadata_type >::reference **metadata_reference**
- typedef Metadata **metadata_type**
- typedef [rebind_traits](#)< _Alloc, [splay_tree_node_](#) >::pointer **node_pointer**
- typedef Value_Type **value_type**

Public Member Functions

- metadata_reference **get_metadata** ()
- metadata_const_reference **get_metadata** () const
- bool **special** () const

Public Attributes

- metadata_type **m_metadata**
- node_pointer **m_p_left**
- node_pointer **m_p_parent**
- node_pointer **m_p_right**
- bool **m_special**
- value_type **m_value**

6.944.1 Detailed Description

```
template<typename Value_Type, class Metadata, typename _Alloc>
struct __gnu_pbds::detail::splay_tree_node_< Value_Type, Metadata, _Alloc >
```

Node for splay tree.

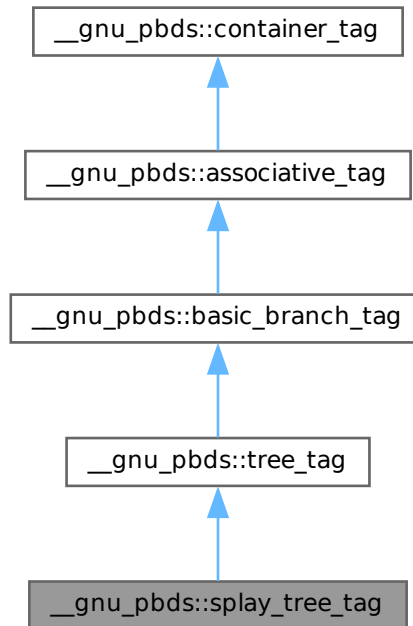
The documentation for this struct was generated from the following file:

- [splay_tree_/node.hpp](#)

6.945 __gnu_pbds::splay_tree_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::splay_tree_tag:



6.945.1 Detailed Description

Splay tree.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.946 std::stack< _Tp, _Sequence > Class Template Reference

```
#include <stl_stack.h>
```

Public Types

- typedef _Sequence::const_reference **const_reference**
- typedef _Sequence **container_type**
- typedef _Sequence::reference **reference**
- typedef _Sequence::size_type **size_type**
- typedef _Sequence::value_type **value_type**

Public Member Functions

- template<typename _Seq = _Sequence, typename _Requires = typename enable_if<is_default_constructible<_Seq>::value>::type>
[stack](#) ()
- **stack** (_Sequence &&__c)

- `template<typename _Alloc, typename _Requires = _Uses<_Alloc>>`
stack (`_Sequence &&__c`, `const _Alloc &__a`)
- `template<typename _Alloc, typename _Requires = _Uses<_Alloc>>`
stack (`const _Alloc &__a`)
- **stack** (`const _Sequence &__c`)
- `template<typename _Alloc, typename _Requires = _Uses<_Alloc>>`
stack (`const _Sequence &__c`, `const _Alloc &__a`)
- `template<typename _Alloc, typename _Requires = _Uses<_Alloc>>`
stack (`const stack &__q`, `const _Alloc &__a`)
- `template<typename _Alloc, typename _Requires = _Uses<_Alloc>>`
stack (`stack &&__q`, `const _Alloc &__a`)
- `template<typename... _Args>`
`decltype(auto)` **emplace** (`_Args &&... __args`)
- `bool empty () const`
- `void pop ()`
- `void push (const value_type &__x)`
- `void push (value_type &&__x)`
- `size_type size () const`
- `void swap (stack &__s) noexcept(__is_nothrow_swappable<_Sequence>::value)`
- `reference top ()`
- `const_reference top () const`

Protected Attributes

- `_Sequence c`

Friends

- `template<typename _Tp1, typename _Seq1 >`
`bool operator< (const stack<_Tp1, _Seq1> &, const stack<_Tp1, _Seq1> &)`
- `template<typename _Tp1, typename _Seq1 >`
`bool operator== (const stack<_Tp1, _Seq1> &, const stack<_Tp1, _Seq1> &)`

6.946.1 Detailed Description

`template<typename _Tp, typename _Sequence = deque<_Tp>>`
class `std::stack<_Tp, _Sequence>`

A standard container giving FILO behavior.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Sequence</code>	Type of underlying sequence, defaults to <code>deque<_Tp></code> .

Meets many of the requirements of a [container](#), but does not define anything to do with iterators. Very few of the other standard container interfaces are defined.

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces strict first-in-last-out stack behavior.

The second template parameter defines the type of the underlying sequence/container. It defaults to `std::deque`, but it can be any type that supports `back`, `push_back`, and `pop_back`, such as `std::list`, `std::vector`, or an appropriate user-defined type.

Members not found in *normal* containers are `container_type`, which is a typedef for the second Sequence param-

eter, and push, pop, and top, which are standard stack/FILO operations.

6.946.2 Constructor & Destructor Documentation

stack()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
template<typename _Seq = _Sequence, typename _Requires = typename enable_if<is_default_constructible<_Seq>::value>::type>
```

```
std::stack< _Tp, _Sequence >::stack ( ) [inline]
```

Default constructor creates no elements.

6.946.3 Member Function Documentation

empty()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
bool std::stack< _Tp, _Sequence >::empty ( ) const [inline]
```

Returns true if the stack is empty.

pop()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
void std::stack< _Tp, _Sequence >::pop ( ) [inline]
```

Removes first element.

This is a typical stack operation. It shrinks the stack by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before pop() is called.

push()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
void std::stack< _Tp, _Sequence >::push (
    const value_type & __x ) [inline]
```

Add data to the top of the stack.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the top of the stack and assigns the given data to it. The time complexity of the operation depends on the underlying sequence.

size()

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
size_type std::stack< _Tp, _Sequence >::size ( ) const [inline]
```

Returns the number of elements in the stack.

top() [1/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
reference std::stack< _Tp, _Sequence >::top ( ) [inline]
```

Returns a read/write reference to the data at the first element of the stack.

top() [2/2]

```
template<typename _Tp , typename _Sequence = deque<_Tp>>
const_reference std::stack< _Tp, _Sequence >::top ( ) const [inline]
```

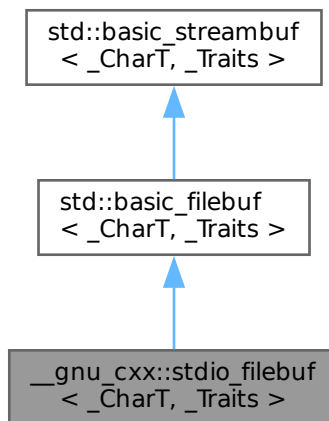
Returns a read-only (constant) reference to the data at the first element of the stack.
The documentation for this class was generated from the following file:

- [std_stack.h](#)

6.947 __gnu_cxx::stdio_filebuf< _CharT, _Traits > Class Template Reference

```
#include <stdio_filebuf.h>
```

Inheritance diagram for `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`:

**Public Types**

- typedef `codecvt< char_type, char, __state_type >` **__codecvt_type**
- typedef `__basic_file< char >` **__file_type**
- typedef `basic_filebuf< char_type, traits_type >` **__filebuf_type**
- typedef `traits_type::state_type` **__state_type**
- typedef `basic_streambuf< char_type, traits_type >` **__streambuf_type**
- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `std::size_t` **size_t**
- typedef `_Traits` **traits_type**

Public Member Functions

- `stdio_filebuf ()`
- `stdio_filebuf (int __fd, std::ios_base::openmode __mode, size_t __size=static_cast< size_t >(BUFSIZ))`
- `stdio_filebuf (std::__c_file * __f, std::ios_base::openmode __mode, size_t __size=static_cast< size_t >(BUFSIZ))`

- `stdio_filebuf` (`stdio_filebuf &&`)=default
- virtual `~stdio_filebuf` ()
- `__filebuf_type` * `close` ()
- int `fd` ()
- `std::__c_file` * `file` ()
- locale `getloc` () const
- streamsize `in_avail` ()
- bool `is_open` () const throw ()
- template<typename `_Path` >
`_If_fs_path<_Path, __filebuf_type * >` `open` (const `_Path` & `__s`, `ios_base::openmode` `__mode`)
- `__filebuf_type` * `open` (const char * `__s`, `ios_base::openmode` `__mode`)
- `__filebuf_type` * `open` (const `std::string` & `__s`, `ios_base::openmode` `__mode`)
- `stdio_filebuf` & `operator=` (`stdio_filebuf &&`)=default
- locale `pubimbue` (const locale & `__loc`)
- int_type `sbumpc` ()
- int_type `sgetc` ()
- streamsize `sgetn` (char_type * `__s`, streamsize `__n`)
- int_type `snextc` ()
- int_type `sputbackc` (char_type `__c`)
- int_type `sputc` (char_type `__c`)
- streamsize `sputn` (const char_type * `__s`, streamsize `__n`)
- int_type `sungetc` ()
- void `swap` (`basic_filebuf` &)
- void `swap` (`stdio_filebuf` & `__fb`)
- `basic_streambuf` * `pubsetbuf` (char_type * `__s`, streamsize `__n`)
- pos_type `pubseekoff` (off_type `__off`, `ios_base::seekdir` `__way`, `ios_base::openmode` `__mode`=`ios_base::in`|`ios_base::out`)
- pos_type `pubseekpos` (pos_type `__sp`, `ios_base::openmode` `__mode`=`ios_base::in`|`ios_base::out`)
- int `pubsync` ()

Protected Member Functions

- void `__safe_gbump` (streamsize `__n`)
- void `__safe_pbump` (streamsize `__n`)
- void `_M_allocate_internal_buffer` ()
- bool `_M_convert_to_external` (char_type *, streamsize)
- void `_M_create_pback` ()
- void `_M_destroy_internal_buffer` () throw ()
- void `_M_destroy_pback` () throw ()
- int `_M_get_ext_pos` (`_state_type` & `__state`)
- pos_type `_M_seek` (off_type `__off`, `ios_base::seekdir` `__way`, `_state_type` `__state`)
- void `_M_set_buffer` (streamsize `__off`)
- bool `_M_terminate_output` ()
- void `gbump` (int `__n`)
- virtual void `imbue` (const locale & `__loc`)
- virtual int_type `overflow` (int_type `__c`=`_Traits::eof`())
- virtual int_type `pbackfail` (int_type `__c`=`_Traits::eof`())
- void `pbump` (int `__n`)
- virtual pos_type `seekoff` (off_type `__off`, `ios_base::seekdir` `__way`, `ios_base::openmode` `__mode`=`ios_base::in`|`ios_base::out`)

- virtual pos_type [seekpos](#) (pos_type __pos, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual [__streambuf_type](#) * [setbuf](#) (char_type * __s, streamsize __n)
- void [setg](#) (char_type * __gbeg, char_type * __gnext, char_type * __gend)
- void [setp](#) (char_type * __pbeg, char_type * __pend)
- virtual streamsize [showmanyc](#) ()
- void [swap](#) ([basic_streambuf](#) & __sb)
- virtual int [sync](#) ()
- virtual int_type [uflow](#) ()
- virtual int_type [underflow](#) ()
- virtual streamsize [xsgetn](#) (char_type * __s, streamsize __n)
- virtual streamsize [xspn](#) (const char_type * __s, streamsize __n)
-
- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const
-
- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- char_type * [_M_buf](#)
- bool [_M_buf_allocated](#)
- locale [_M_buf_locale](#)
- size_t [_M_buf_size](#)
- const [__codecvt_type](#) * [_M_codecvt](#)
- char * [_M_ext_buf](#)
- streamsize [_M_ext_buf_size](#)
- char * [_M_ext_end](#)
- const char * [_M_ext_next](#)
- [__file_type](#) [_M_file](#)
- char_type * [_M_in_beg](#)
- char_type * [_M_in_cur](#)
- char_type * [_M_in_end](#)
- [__c_lock](#) [_M_lock](#)
- ios_base::openmode [_M_mode](#)
- char_type * [_M_out_beg](#)
- char_type * [_M_out_cur](#)
- char_type * [_M_out_end](#)
- bool [_M_reading](#)
- [__state_type](#) [_M_state_beg](#)
- [__state_type](#) [_M_state_cur](#)
- [__state_type](#) [_M_state_last](#)
- bool [_M_writing](#)
-
- char_type [_M_pback](#)
- char_type * [_M_pback_cur_save](#)
- char_type * [_M_pback_end_save](#)
- bool [_M_pback_init](#)

6.947.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class __gnu_cxx::stdio_filebuf< _CharT, _Traits >
```

Provides a layer of compatibility for C/POSIX.

This GNU extension provides extensions for working with standard C FILE*'s and POSIX file descriptors. It must be instantiated by the user with the type of character used in the file stream, e.g., `stdio_filebuf<char>`.

6.947.2 Constructor & Destructor Documentation

`stdio_filebuf()` [1/3]

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf ( ) [inline]
deferred initialization
```

`stdio_filebuf()` [2/3]

```
template<typename _CharT , typename _Traits >
__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf (
    int __fd,
    std::ios_base::openmode __mode,
    size_t __size = static_cast<size_t>( BUFSIZ ) )
```

Parameters

<code>__fd</code>	An open file descriptor.
<code>__mode</code>	Same meaning as in a standard filebuf.
<code>__size</code>	Optimal or preferred size of internal buffer, in chars.

This constructor associates a file stream buffer with an open POSIX file descriptor. The file descriptor will be automatically closed when the `stdio_filebuf` is closed/destroyed.

`stdio_filebuf()` [3/3]

```
template<typename _CharT , typename _Traits >
__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf (
    std::__c_file * __f,
    std::ios_base::openmode __mode,
    size_t __size = static_cast<size_t>( BUFSIZ ) )
```

Parameters

<code>__f</code>	An open FILE*.
<code>__mode</code>	Same meaning as in a standard filebuf.
<code>__size</code>	Optimal or preferred size of internal buffer, in chars. Defaults to system's BUFSIZ.

This constructor associates a file stream buffer with an open C FILE*. The FILE* will not be automatically closed when the `stdio_filebuf` is closed/destroyed.

`~stdio_filebuf()`

```
template<typename _CharT , typename _Traits >
__gnu_cxx::stdio_filebuf< _CharT, _Traits >::~~stdio_filebuf [virtual]
```

Closes the external data stream if the file descriptor constructor was used.

6.947.3 Member Function Documentation

_M_create_pback()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_create_pback ( ) [inline], [protected], [inherited]
```

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

_M_destroy_pback()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_destroy_pback ( ) throw ( ) [inline], [protected],
[inherited]
```

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

_M_set_buffer()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::_M_set_buffer (
    streamsize __off ) [inline], [protected], [inherited]
```

This function sets the pointers of the internal buffer, both get and put areas. Typically:

`__off == egptr() - eback()` upon underflow/uflow (**read** mode); `__off == 0` upon overflow (**write** mode); `__off == -1` upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: `egptr() - pbase() == _M_buf_size - 1`, since `_M_buf_size` reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

Referenced by `std::basic_filebuf< _CharT, _Traits >::close()`.

close()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::__filebuf_type * std::basic_filebuf< _CharT, _Traits >::close
[inherited]
```

Closes the currently associated file.

Returns

this on success, NULL on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

References `std::basic_filebuf< _CharT, _Traits >::_M_mode`, `std::basic_filebuf< _CharT, _Traits >::_M_pback_init`, `std::basic_filebuf< _CharT, _Traits >::_M_reading`, and `std::basic_filebuf< _CharT, _Traits >::_M_set_buffer()`.

eback()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence

- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf<\_CharT, \_Traits>::egptr ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow\(\)](#).

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf<\_CharT, \_Traits>::epptr ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

fd()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
int \_\_gnu\_cxx::stdio\_filebuf<\_CharT, \_Traits>::fd ( ) [inline]
```

Returns

The underlying file descriptor.

Once associated with an external data stream, this function can be used to access the underlying POSIX file descriptor. Note that there is no way for the library to track what you do with the descriptor, so be careful.

file()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
std::_c_file * \_\_gnu\_cxx::stdio\_filebuf<\_CharT, \_Traits>::file ( ) [inline]
```

Returns

The underlying FILE*.

This function can be used to access the underlying "C" file pointer. Note that there is no way for the library to track what you do with the file, so be careful.

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow()`.

imbue()

```
template<typename _CharT , typename _Traits >
void std::basic_filebuf< _CharT, _Traits >::imbue (
    const locale & __loc ) [protected], [virtual], [inherited]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::cur`.

in_avail()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ( ) [inline], [inherited]
Looking ahead into the stream.
```

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

is_open()

```
template<typename _CharT, typename _Traits>
bool std::basic_filebuf<_CharT, _Traits>::is_open ( ) const throw ( ) [inline], [inherited]
Returns true if the external file is open.
```

open() [1/3]

```
template<typename _CharT, typename _Traits>
template<typename _Path>
_If_fs_path<_Path, __filebuf_type * > std::basic_filebuf<_CharT, _Traits>::open (
    const _Path & __s,
    ios_base::openmode __mode ) [inline], [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file, as a <code>filesystem::path</code> .
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

open() [2/3]

```
template<typename _CharT, typename _Traits>
basic_filebuf<_CharT, _Traits>::__filebuf_type * std::basic_filebuf<_CharT, _Traits>::open (
    const char * __s,
    ios_base::openmode __mode ) [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines app, in|out|app, in|app, binary|app, binary|in|out|app, and binary|in|app per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r
	+	+			r+
	+	+	+		w+
	+	+		+	a+
	+			+	a+
+		+			wb
+		+		+	ab
+				+	ab
+		+	+		wb
+	+				rb
+	+	+			r+b
+	+	+	+		w+b
+	+	+		+	a+b
+	+			+	a+b

References [std::ios_base::ate](#), [std::ios_base::end](#), and [std::basic_filebuf<_CharT, _Traits>::open\(\)](#).

Referenced by [std::basic_filebuf<_CharT, _Traits>::open\(\)](#).

open() [3/3]

```
template<typename _CharT , typename _Traits >
__filebuf_type * std::basic_filebuf< _CharT, _Traits >::open (
    const std::string & __s,
    ios_base::openmode __mode ) [inline], [inherited]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

overflow()

```
template<typename _CharT , typename _Traits >
```

```
basic_filebuf<_CharT, _Traits>::int_type std::basic_filebuf<_CharT, _Traits>::overflow (
    int_type __c = _Traits::eof() ) [protected], [virtual], [inherited]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::app`, `std::ios_base::cur`, and `std::ios_base::out`.

pbackfail()

```
template<typename _CharT, typename _Traits>
basic_filebuf<_CharT, _Traits>::int_type std::basic_filebuf<_CharT, _Traits>::pbackfail (
    int_type __c = _Traits::eof() ) [protected], [virtual], [inherited]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::cur`, and `std::ios_base::in`.

pbase()

```
template<typename _CharT, typename _Traits>
char_type * std::basic_streambuf<_CharT, _Traits>::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline], [inherited]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekoff` function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual `sync` function.

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::sync()`, and `std::basic_istream<_CharT, _Traits>::sync()`.

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Referenced by [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< char >::ignore\(\)](#), [std::istreambuf_iterator< _CharT, _Traits >::operator++\(\)](#), and [std::basic_istream< char >::seekg\(\)](#).

seekoff()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf< _CharT, _Traits >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual], [inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

References [std::ios_base::cur](#).

seekpos()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf< _CharT, _Traits >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [protected], [virtual], [inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

References [std::ios_base::beg](#).

setbuf()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::__streambuf_type * std::basic_filebuf< _CharT, _Traits >↵
::setbuf (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

setg()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert()`.

setp()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setp (
    char_type * __pbeg,
    char_type * __pend) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Referenced by `std::wbuffer_convert<_Codecv, _Elem, _Tr>::wbuffer_convert()`.

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::istreambuf_iterator<_CharT, _Traits>::operator++()`, and `std::basic_istream<char>::seekg()`.

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for `xsggetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsggetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_filebuf<_CharT, _Traits>::showmanyc [protected], [virtual], [inherited]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to underflow or uflow] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::binary`, and `std::ios_base::in`.

snextc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::snextc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::seekg()`, and `std::basic_istream<char>::unget()`.

sputbackc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
    char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

__c, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores __c in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(↵__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::sputn (
    const char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>↵__s</code>	A buffer read area.
<code>↵__n</code>	A count.

One of two public output functions.

Returns `xsputn(__s,__n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

sungetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sungetc ( ) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbckfail()`. The effect is to *unget* the last character *gotten*.

Referenced by `std::basic_istream<_CharT, _Traits>::ungetc()`.

sync()

```
template<typename _CharT , typename _Traits >
int std::basic_filebuf< _CharT, _Traits >::sync [protected], [virtual], [inherited]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

uflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual],
[inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

underflow()

```
template<typename _CharT , typename _Traits >
basic_filebuf< _CharT, _Traits >::int_type std::basic_filebuf< _CharT, _Traits >::underflow
[protected], [virtual], [inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::ios_base::in`.

xsgetn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_filebuf< _CharT, _Traits >::xsgetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

References [std::ios_base::in](#).

xspn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_filebuf<_CharT, _Traits>::xspn (
    const char_type * __s,
    streamsize __n) [protected], [virtual], [inherited]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

References [std::ios_base::app](#), [std::min\(\)](#), and [std::ios_base::out](#).

6.947.4 Member Data Documentation**M_buf**

```
template<typename _CharT, typename _Traits>
char_type* std::basic_filebuf<_CharT, _Traits>::M_buf [protected], [inherited]
```

Pointer to the beginning of internal buffer.

M_buf_locale

```
template<typename _CharT, typename _Traits>
locale std::basic_streambuf<_CharT, _Traits>::M_buf_locale [protected], [inherited]
```

Current locale setting.

Referenced by [std::basic_filebuf<_CharT, _Traits>::basic_filebuf\(\)](#).

M_buf_size

```
template<typename _CharT, typename _Traits>
size_t std::basic_filebuf<_CharT, _Traits>::M_buf_size [protected], [inherited]
```

Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.

`_M_ext_buf`

```
template<typename _CharT , typename _Traits >
```

```
char* std::basic\_filebuf< _CharT, _Traits >::_M_ext_buf [protected], [inherited]
```

Buffer for external characters. Used for input when `codecvt::always_noconv() == false`. When valid, this corresponds to `eback()`.

`_M_ext_buf_size`

```
template<typename _CharT , typename _Traits >
```

```
streamsize std::basic\_filebuf< _CharT, _Traits >::_M_ext_buf_size [protected], [inherited]
```

Size of buffer held by `_M_ext_buf`.

`_M_ext_next`

```
template<typename _CharT , typename _Traits >
```

```
const char* std::basic\_filebuf< _CharT, _Traits >::_M_ext_next [protected], [inherited]
```

Pointers into the buffer held by `_M_ext_buf` that delimit a subsequence of bytes that have been read but not yet converted. When valid, `_M_ext_next` corresponds to `egptr()`.

`_M_in_beg`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_in_beg [protected], [inherited]
```

Start of get area.

`_M_in_cur`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_in_cur [protected], [inherited]
```

Current read area.

`_M_in_end`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_in_end [protected], [inherited]
```

End of get area.

`_M_mode`

```
template<typename _CharT , typename _Traits >
```

```
ios_base::openmode std::basic\_filebuf< _CharT, _Traits >::_M_mode [protected], [inherited]
```

Place to stash in || out || in | out settings for current filebuf.

Referenced by [std::basic_filebuf<_CharT, _Traits>::close\(\)](#).

`_M_out_beg`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_out_beg [protected], [inherited]
```

Start of put area.

`_M_out_cur`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_out_cur [protected], [inherited]
```

Current put area.

`_M_out_end`

```
template<typename _CharT , typename _Traits >
char_type* std::basic\_streambuf< _CharT, _Traits >::_M_out_end [protected], [inherited]
End of put area.
```

`_M_pback`

```
template<typename _CharT , typename _Traits >
char_type std::basic\_filebuf< _CharT, _Traits >::_M_pback [protected], [inherited]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

`_M_pback_cur_save`

```
template<typename _CharT , typename _Traits >
char_type* std::basic\_filebuf< _CharT, _Traits >::_M_pback_cur_save [protected], [inherited]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

`_M_pback_end_save`

```
template<typename _CharT , typename _Traits >
char_type* std::basic\_filebuf< _CharT, _Traits >::_M_pback_end_save [protected], [inherited]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

`_M_pback_init`

```
template<typename _CharT , typename _Traits >
bool std::basic\_filebuf< _CharT, _Traits >::_M_pback_init [protected], [inherited]
Necessary bits for putback buffer management.
```

Note

pbacks of over one character are not currently supported.

Referenced by [std::basic_filebuf< _CharT, _Traits >::close\(\)](#).

`_M_reading`

```
template<typename _CharT , typename _Traits >
bool std::basic\_filebuf< _CharT, _Traits >::_M_reading [protected], [inherited]
_M_reading == false && _M_writing == false for uncommitted mode; _M_reading == true for read mode; _M_writing
== true for write mode;
```

NB: _M_reading == true && _M_writing == true is unused.

Referenced by [std::basic_filebuf< _CharT, _Traits >::close\(\)](#).

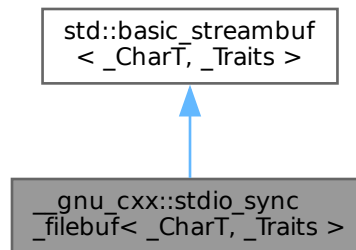
The documentation for this class was generated from the following file:

- [stdio_filebuf.h](#)

6.948 `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>` Class Template Reference

```
#include <stdio_sync_filebuf.h>
```

Inheritance diagram for `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`:

**Public Types**

- typedef `_CharT` **char_type**
- typedef traits_type::int_type **int_type**
- typedef traits_type::off_type **off_type**
- typedef traits_type::pos_type **pos_type**
- typedef `_Traits` **traits_type**

Public Member Functions

- **stdio_sync_filebuf** (std::__c_file * __f)
- **stdio_sync_filebuf** (stdio_sync_filebuf && __fb) noexcept
- std::__c_file * **file** ()
- locale **getloc** () const
- streamsize **in_avail** ()
- **stdio_sync_filebuf** & **operator=** (stdio_sync_filebuf && __fb) noexcept
- locale **pubimbue** (const locale & __loc)
- int_type **sbumpc** ()
- int_type **sgetc** ()
- streamsize **sgetn** (char_type * __s, streamsize __n)
- int_type **snextc** ()
- int_type **sputbackc** (char_type __c)
- int_type **sputc** (char_type __c)
- streamsize **sputn** (const char_type * __s, streamsize __n)
- int_type **sungetc** ()
- void **swap** (stdio_sync_filebuf & __fb)
- **basic_streambuf** * **pubsetbuf** (char_type * __s, streamsize __n)
- pos_type **pubseekoff** (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- pos_type **pubseekpos** (pos_type __sp, ios_base::openmode __mode=ios_base::in|ios_base::out)
- int **pubsync** ()

Protected Member Functions

- void **__safe_gbump** (streamsize __n)
 - void **__safe_pbump** (streamsize __n)
 - void **gbump** (int __n)
 - virtual void **imbue** (const locale &__loc)
 - virtual int_type **overflow** (int_type __c=traits_type::eof())
 - virtual int_type **pbackfail** (int_type __c=traits_type::eof())
 - void **pbump** (int __n)
 - virtual pos_type **seekoff** (off_type, ios_base::seekdir, ios_base::openmode=ios_base::in|ios_base::out)
 - virtual **std::streampos seekoff** (**std::streamoff** __off, **std::ios_base::seekdir** __dir, **std::ios_base::openmode**=**std::ios_base::in**|**std::ios_base::out**)
 - virtual pos_type **seekpos** (pos_type, ios_base::openmode=ios_base::in|ios_base::out)
 - virtual **std::streampos seekpos** (**std::streampos** __pos, **std::ios_base::openmode** __mode=**std::ios_base::in**|**std::ios_base::out**)
 - virtual **basic_streambuf**< char_type, _Traits > * **setbuf** (char_type *, streamsize)
 - void **setg** (char_type * __gbeg, char_type * __gnext, char_type * __gend)
 - void **setp** (char_type * __pbeg, char_type * __pend)
 - virtual streamsize **showmanyc** ()
 - void **swap** (**basic_streambuf** & __sb)
 - virtual int **sync** ()
 - int_type **syncgetc** ()
 - **stdio_sync_filebuf**< char >::int_type **syncgetc** ()
 - **stdio_sync_filebuf**< wchar_t >::int_type **syncgetc** ()
 - int_type **syncputc** (int_type __c)
 - **stdio_sync_filebuf**< char >::int_type **syncputc** (int_type __c)
 - **stdio_sync_filebuf**< wchar_t >::int_type **syncputc** (int_type __c)
 - int_type **syncungetc** (int_type __c)
 - **stdio_sync_filebuf**< char >::int_type **syncungetc** (int_type __c)
 - **stdio_sync_filebuf**< wchar_t >::int_type **syncungetc** (int_type __c)
 - virtual int_type **uflow** ()
 - virtual int_type **underflow** ()
 - **std::streamsize xsgetn** (char *__s, **std::streamsize** __n)
 - virtual **std::streamsize xsgetn** (char_type *__s, **std::streamsize** __n)
 - virtual streamsize **xsgetn** (char_type *__s, streamsize __n)
 - **std::streamsize xsgetn** (wchar_t *__s, **std::streamsize** __n)
 - **std::streamsize xspun** (const char *__s, **std::streamsize** __n)
 - virtual **std::streamsize xspun** (const char_type *__s, **std::streamsize** __n)
 - virtual streamsize **xspun** (const char_type *__s, streamsize __n)
 - **std::streamsize xspun** (const wchar_t *__s, **std::streamsize** __n)
-
- char_type * **eback** () const
 - char_type * **gptr** () const
 - char_type * **egptr** () const
-
- char_type * **pbase** () const
 - char_type * **pptr** () const
 - char_type * **eptr** () const

Protected Attributes

- locale [_M_buf_locale](#)
- char_type * [_M_in_beg](#)
- char_type * [_M_in_cur](#)
- char_type * [_M_in_end](#)
- char_type * [_M_out_beg](#)
- char_type * [_M_out_cur](#)
- char_type * [_M_out_end](#)

6.948.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class __gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>
```

Provides a layer of compatibility for C.

This GNU extension provides extensions for working with standard C FILE*'s. It must be instantiated by the user with the type of character used in the file stream, e.g., `stdio_filebuf<char>`.

6.948.2 Member Function Documentation

eback()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf<\_CharT, \_Traits>::eback \( \) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

egptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic\_streambuf<\_CharT, \_Traits>::egptr \( \) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow\(\)](#).

epptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- epptr() returns the end pointer for the output sequence

file()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
std::__c_file * __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::file ( ) [inline]
```

Returns

The underlying FILE*.

This function can be used to access the underlying C file pointer. Note that there is no way for the library to track what you do with the file, so be careful.

gbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

The current locale in effect.

If pubimbue(loc) has been called, then the most recent loc is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert<_Codecv, _Elem, _Tr>::underflow()`.

`imbue()`

```
template<typename _CharT, typename _Traits>
virtual void std::basic_streambuf<_CharT, _Traits>::imbue (
    const locale & __loc) [inline], [protected], [virtual], [inherited]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

`in_avail()`

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ( ) [inline], [inherited]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

`overflow()`

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
virtual int_type __gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::overflow (
    int_type __c = traits_type::eof() ) [inline], [protected], [virtual]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

pbackfail()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
virtual int_type __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::pbackfail (
    int_type __c = traits_type::eof() ) [inline], [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline], [inherited]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
--------------------	---------

Parameters

<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekoff function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekpos function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual sync function.

Referenced by `std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Referenced by `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< char >::ignore()`, `std::istreambuf_iterator< _CharT, _Traits >::operator++()`, and `std::basic_istream< char >::seekg()`.

seekoff()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, traits_type, _Alloc>`.

seekpos()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, traits_type, _Alloc>`.

setbuf()

```
template<typename _CharT , typename _Traits >
virtual basic_streambuf< char_type, _Traits > * std::basic_streambuf< _CharT, _Traits >::setbuf
(
    char_type * ,
    streamsize ) [inline], [protected], [virtual], [inherited]
```

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, traits_type, _Alloc>`.

setg()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setg (
    char_type * __gbeg,
```

```
char_type * __gnext,
char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

```
__gbeg == eback(), __gnext == gptr(), and __gend == egptr()
```

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

setp()

```
template<typename _CharT, typename _Traits>
void std::basic_streambuf<_CharT, _Traits>::setp (
    char_type * __pbeg,
    char_type * __pend ) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

```
__pbeg == pbase(), __pbeg == pptr(), and __pend == ep_ptr()
```

Referenced by [std::wbuffer_convert<_Codecvt, _Elem, _Tr>::wbuffer_convert\(\)](#).

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<_CharT, _Traits>::getline\(\)](#), [std::basic_istream<_CharT, _Traits>::ignore\(\)](#), [std::istreambuf_iterator<_CharT, _Traits>::operator++\(\)](#), and [std::basic_istream<char>::seekg\(\)](#).

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
```

```
char_type * __s,
streamsize __n ) [inline], [inherited]
```

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

`showmanyc()`

```
template<typename _CharT, typename _Traits>
virtual streamsize std::basic_streambuf<_CharT, _Traits>::showmanyc ( ) [inline], [protected],
[virtual], [inherited]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, _Traits>`.

`snextc()`

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::snextc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

Calls `sgetc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<char>::seekg()`, and `std::basic_istream<char>::unget()`.

`sputbackc()`

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

sputc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sputn (
    const char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xputn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

sungetc()

```
template<typename _CharT , typename _Traits >
int_type std::basic\_streambuf<\_CharT, \_Traits>::sungetc \( \) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Referenced by [std::basic_istream<_CharT, _Traits>::unget\(\)](#).

sync()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
virtual int \_\_gnu\_cxx::stdio\_sync\_filebuf<\_CharT, \_Traits>::sync \( \) [inline], [protected],
[virtual]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

uflow()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
virtual int_type \_\_gnu\_cxx::stdio\_sync\_filebuf<\_CharT, \_Traits>::uflow \( \) [inline], [protected],
[virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

underflow()

```
template<typename _CharT , typename _Traits = std::char_traits<_CharT>>
virtual int_type \_\_gnu\_cxx::stdio\_sync\_filebuf<\_CharT, \_Traits>::underflow \( \) [inline], [protected],
[virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

xsggetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::xsggetn (
    char_type * __s,
    streamsize __n) [protected], [virtual], [inherited]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

References `std::min()`.

xsputn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::xsputn (
    const char_type * __s,
    streamsize __n) [protected], [virtual], [inherited]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either *n* characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

References `std::min()`.

6.948.3 Member Data Documentation

`_M_buf_locale`

```
template<typename _CharT, typename _Traits>
```

```
locale std::basic\_streambuf<\_CharT, \_Traits>::\_M\_buf\_locale [protected], [inherited]
```

Current locale setting.

Referenced by [std::basic_filebuf<_CharT, _Traits>::basic_filebuf\(\)](#).

`_M_in_beg`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_in\_beg [protected], [inherited]
```

Start of get area.

`_M_in_cur`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_in\_cur [protected], [inherited]
```

Current read area.

`_M_in_end`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_in\_end [protected], [inherited]
```

End of get area.

`_M_out_beg`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_out\_beg [protected], [inherited]
```

Start of put area.

`_M_out_cur`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_out\_cur [protected], [inherited]
```

Current put area.

`_M_out_end`

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic\_streambuf<\_CharT, \_Traits>::\_M\_out\_end [protected], [inherited]
```

End of put area.

The documentation for this class was generated from the following file:

- [stdio_sync_filebuf.h](#)

6.949 `std::chrono::_V2::steady_clock` Struct Reference

```
#include <chrono.h>
```

Public Types

- typedef [chrono::nanoseconds](#) **duration**
- typedef duration::period **period**
- typedef duration::rep **rep**
- typedef [chrono::time_point](#)< [steady_clock](#), [duration](#) > **time_point**

Static Public Member Functions

- static [time_point](#) **now** () noexcept

Static Public Attributes

- static constexpr bool **is_steady**

6.949.1 Detailed Description

Monotonic clock.

Time returned has the property of only increasing at a uniform rate.

The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.950 `std::stop_callback<_Callback>` Class Template Reference

Public Types

- using **callback_type** = `_Callback`

Public Member Functions

- **stop_callback** (const [stop_callback](#) &)=delete
- template<typename `_Cb` , [enable_if_t](#)< is_constructible_v< `_Callback`, `_Cb` >, int > = 0>
stop_callback (const [stop_token](#) &__token, `_Cb` &&__cb) noexcept(is_nothrow_constructible_v< `_Callback`, `↔`
`_Cb` >)
- **stop_callback** ([stop_callback](#) &&)=delete
- template<typename `_Cb` , [enable_if_t](#)< is_constructible_v< `_Callback`, `_Cb` >, int > = 0>
stop_callback ([stop_token](#) &&__token, `_Cb` &&__cb) noexcept(is_nothrow_constructible_v< `_Callback`, `_Cb` >)
- [stop_callback](#) & **operator=** (const [stop_callback](#) &)=delete
- [stop_callback](#) & **operator=** ([stop_callback](#) &&)=delete

6.950.1 Detailed Description

```
template<typename _Callback>  
class std::stop_callback< _Callback >
```

A wrapper for callbacks to be run when a stop request is made.

The documentation for this class was generated from the following file:

- [stop_token](#)

6.951 `std::stop_source` Class Reference

Public Member Functions

- `stop_source` (const `stop_source` &__other) noexcept
- `stop_source` (`std::nostopstate_t`) noexcept
- `stop_source` (`stop_source` &&) noexcept=default
- `stop_token` `get_token` () const noexcept
- `stop_source` & `operator=` (const `stop_source` &__other) noexcept
- `stop_source` & `operator=` (`stop_source` &&) noexcept=default
- bool `request_stop` () const noexcept
- bool `stop_possible` () const noexcept
- bool `stop_requested` () const noexcept
- void `swap` (`stop_source` &__other) noexcept

Friends

- bool `operator==` (const `stop_source` &__a, const `stop_source` &__b) noexcept
- void `swap` (`stop_source` &__lhs, `stop_source` &__rhs) noexcept

6.951.1 Detailed Description

A type that allows a stop request to be made.

The documentation for this class was generated from the following file:

- `stop_token`

6.952 `std::stop_token` Class Reference

Public Member Functions

- `stop_token` (const `stop_token` &) noexcept=default
- `stop_token` (`stop_token` &&) noexcept=default
- `stop_token` & `operator=` (const `stop_token` &) noexcept=default
- `stop_token` & `operator=` (`stop_token` &&) noexcept=default
- bool `stop_possible` () const noexcept
- bool `stop_requested` () const noexcept
- void `swap` (`stop_token` &__rhs) noexcept

Friends

- bool `operator==` (const `stop_token` &__a, const `stop_token` &__b)
- template<typename _Callback >
class `stop_callback`
- class `stop_source`
- void `swap` (`stop_token` &__lhs, `stop_token` &__rhs) noexcept

6.952.1 Detailed Description

Allow testing whether a stop request has been made on a `stop_source`.

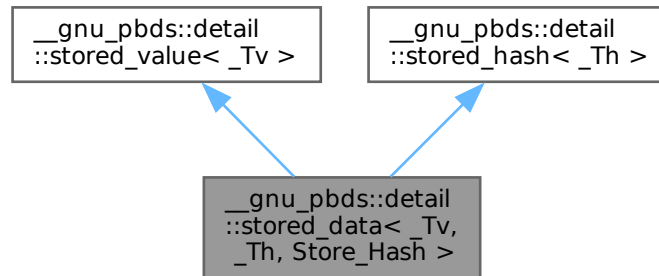
The documentation for this class was generated from the following file:

- `stop_token`

6.953 `__gnu_pbds::detail::stored_data<_Tv,_Th,Store_Hash>` Struct Template Reference

```
#include <types_traits.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::stored_data<_Tv,_Th,Store_Hash>`:



Public Types

- typedef `_Th` **hash_type**
- typedef `_Tv` **value_type**

Public Attributes

- hash_type **m_hash**
- value_type **m_value**

6.953.1 Detailed Description

```
template<typename _Tv, typename _Th, bool Store_Hash>
struct __gnu_pbds::detail::stored_data<_Tv,_Th,Store_Hash>
```

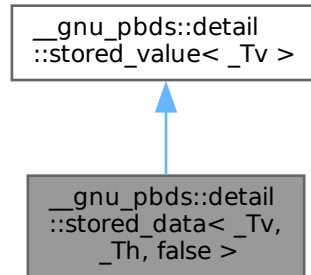
Primary template for representation of stored data. Two types of data can be stored: value and hash.
The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.954 `__gnu_pbds::detail::stored_data<_Tv,_Th,false>` Struct Template Reference

```
#include <types_traits.hpp>
```

Inheritance diagram for __gnu_pbds::detail::stored_data< _Tv, _Th, false >:



Public Types

- typedef `_Tv` `value_type`

Public Attributes

- `value_type` `m_value`

6.954.1 Detailed Description

```
template<typename _Tv, typename _Th>
struct __gnu_pbds::detail::stored_data< _Tv, _Th, false >
```

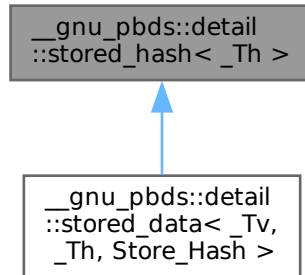
Specialization for representation of stored data of just value type.
The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.955 __gnu_pbds::detail::stored_hash< _Th > Struct Template Reference

```
#include <types_traits.hpp>
```


Inheritance diagram for `__gnu_pbds::detail::stored_hash<_Th>`:



Public Types

- typedef `_Th` **hash_type**

Public Attributes

- hash_type **m_hash**

6.955.1 Detailed Description

```
template<typename _Th>
struct __gnu_pbds::detail::stored_hash<_Th>
```

Stored hash.

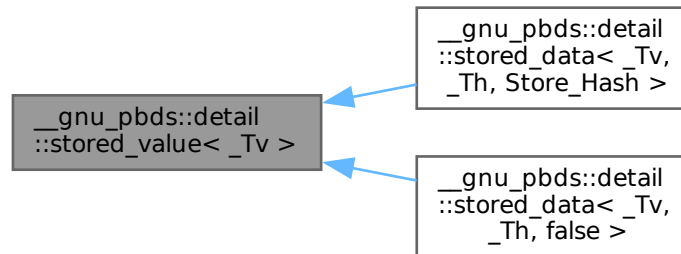
The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.956 `__gnu_pbds::detail::stored_value<_Tv>` Struct Template Reference

```
#include <types_traits.hpp>
```

Inheritance diagram for __gnu_pbds::detail::stored_value<_Tv>:



Public Types

- typedef `_Tv` **value_type**

Public Attributes

- value_type **m_value**

6.956.1 Detailed Description

```
template<typename _Tv>
struct __gnu_pbds::detail::stored_value<_Tv>
```

Stored value.

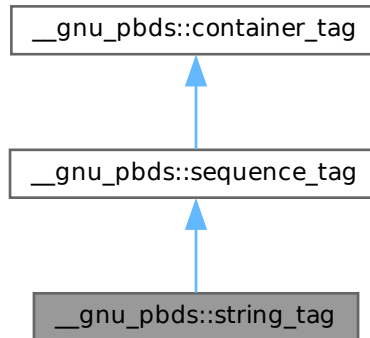
The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.957 __gnu_pbds::string_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::string_tag`:



6.957.1 Detailed Description

Basic string container, inclusive of strings, ropes, etc.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.958 `std::student_t_distribution<_RealType>` Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- **`student_t_distribution`** (`_RealType __n`)
- **`student_t_distribution`** (const [param_type](#) &__p)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void **`generate`** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void **`generate`** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `template<typename _UniformRandomNumberGenerator >`
void **`generate`** ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator &__urng`)
- `template<typename _UniformRandomNumberGenerator >`
void **`generate`** ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator &__urng, const param_type &__p`)
- [result_type](#) **`max`** () const

- [result_type](#) min () const
- [_RealType](#) n () const
- template<typename [_UniformRandomNumberGenerator](#) >
[result_type](#) operator() ([_UniformRandomNumberGenerator](#) &__urng)
- template<typename [_UniformRandomNumberGenerator](#) >
[result_type](#) operator() ([_UniformRandomNumberGenerator](#) &__urng, const [param_type](#) &__p)
- [param_type](#) param () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- template<typename [_RealType1](#) , typename [_CharT](#) , typename [_Traits](#) >
[std::basic_ostream](#)< [_CharT](#), [_Traits](#) > & [operator<<](#) ([std::basic_ostream](#)< [_CharT](#), [_Traits](#) > &__os, const [std::student_t_distribution](#)< [_RealType1](#) > &__x)
- bool [operator==](#) (const [student_t_distribution](#) &__d1, const [student_t_distribution](#) &__d2)
- template<typename [_RealType1](#) , typename [_CharT](#) , typename [_Traits](#) >
[std::basic_istream](#)< [_CharT](#), [_Traits](#) > & [operator>>](#) ([std::basic_istream](#)< [_CharT](#), [_Traits](#) > &__is, [std::student_t_distribution](#)< [_RealType1](#) > &__x)

6.958.1 Detailed Description

template<typename [_RealType](#) = double>
class [std::student_t_distribution](#)< [_RealType](#) >

A [student_t_distribution](#) random number distribution.

The formula for the normal probability mass function is:

$$p(x|n) = \frac{1}{\sqrt{(n\pi)}} \frac{\Gamma((n+1)/2)}{\Gamma(n/2)} \left(1 + \frac{x^2}{n}\right)^{-(n+1)/2}$$

6.958.2 Member Typedef Documentation

result_type

template<typename [_RealType](#) = double>
typedef [_RealType](#) [std::student_t_distribution](#)< [_RealType](#) >::result_type
The type of the range of the distribution.

6.958.3 Member Function Documentation

max()

template<typename [_RealType](#) = double>
[result_type](#) [std::student_t_distribution](#)< [_RealType](#) >::max () const [inline]
Returns the least upper bound value of the distribution.
References [std::numeric_limits](#)< [_Tp](#) >::max().

min()

template<typename [_RealType](#) = double>
[result_type](#) [std::student_t_distribution](#)< [_RealType](#) >::min () const [inline]
Returns the greatest lower bound value of the distribution.
References [std::numeric_limits](#)< [_Tp](#) >::lowest().

operator()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::student_t_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::sqrt\(\)](#).

param() [1/2]

```
template<typename _RealType = double>
param_type std::student_t_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

param() [2/2]

```
template<typename _RealType = double>
void std::student_t_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::student_t_distribution< _RealType >::reset ( ) [inline]
```

Resets the distribution state.

References [std::normal_distribution< _RealType >::reset\(\)](#), and [std::gamma_distribution< _RealType >::reset\(\)](#).

6.958.4 Friends And Related Symbol Documentation**operator<<**

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::student_t_distribution< _RealType1 > & __x ) [friend]
```

Inserts a `student_t_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>student_t_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _RealType = double>
bool operator== (
    const student_t_distribution< _RealType > & __d1,
    const student_t_distribution< _RealType > & __d2 ) [friend]
```

Return true if two Student t distributions have the same parameters and the sequences that would be generated are equal.

operator>>

```
template<typename _RealType = double>
template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_istream< _CharT, _Traits > & operator>> (
    std::basic_istream< _CharT, _Traits > & __is,
    std::student_t_distribution< _RealType1 > & __x ) [friend]
```

Extracts a student_t_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A student_t_distribution random number generator engine.

Returns

The input stream with __x extracted or in an error state.

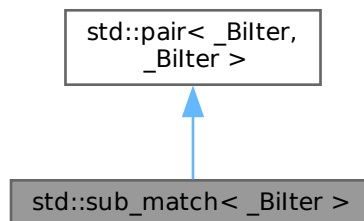
The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.959 std::sub_match<_Bilter > Class Template Reference

```
#include <regex.h>
```

Inheritance diagram for std::sub_match<_Bilter >:



Public Types

- typedef `__iter_traits::difference_type` **difference_type**
- typedef `_Bilter` **first_type**
- typedef `_Bilter` **iterator**
- typedef `_Bilter` **second_type**
- typedef `basic_string< value_type >` **string_type**
- typedef `__iter_traits::value_type` **value_type**

Public Member Functions

- int **compare** (const `sub_match` &__s) const
- `difference_type` **length** () const noexcept
- `operator string_type` () const
- `string_type` **str** () const
- constexpr void **swap** (`pair` &__p) noexcept(__and< __is_nothrow_swappable< `_Bilter` >, __is_nothrow_swappable< `_Bilter` >>::value)
- void **swap** (`sub_match` &__s) noexcept(__is_nothrow_swappable< `_Bilter` >::value)
- int **compare** (const `string_type` &__s) const
- int **compare** (const `value_type` *__s) const

Public Attributes

- `_Bilter` **first**
- bool **matched**
- `_Bilter` **second**

Related Symbols

(Note that these are not member symbols.)

- constexpr `pair< typename __decay_and_strip< _Bilter >::__type, typename __decay_and_strip< _Bilter >::__type >` **make_pair** (`_Bilter` &&__x, `_Bilter` &&__y)
- template<typename `_Bilter` >
bool **operator==** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bilter` >
bool **operator!=** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bilter` >
bool **operator<** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bilter` >
bool **operator<=** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bilter` >
bool **operator>=** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bilter` >
bool **operator>** (const `sub_match`< `_Bilter` > &__lhs, const `sub_match`< `_Bilter` > &__rhs)
- template<typename `_Bi_iter` , typename `_Ch_traits` , typename `_Ch_alloc` >
bool **operator==** (const `__sub_match_string`< `_Bi_iter`, `_Ch_traits`, `_Ch_alloc` > &__lhs, const `sub_match`< `_Bi_iter` > &__rhs)
- template<typename `_Bi_iter` , typename `_Ch_traits` , typename `_Ch_alloc` >
bool **operator!=** (const `__sub_match_string`< `_Bi_iter`, `_Ch_traits`, `_Ch_alloc` > &__lhs, const `sub_match`< `_Bi_iter` > &__rhs)

- Generated by Doxygen

- `template<typename _Bi_iter >`
`bool operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool operator== (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator!= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator< (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator> (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator>= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator<= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator!= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter >`
`basic_ostream< _Ch_type, _Ch_traits > & operator<< (basic_ostream< _Ch_type, _Ch_traits > &__os, const sub_match< _Bi_iter > &__m)`

- constexpr [enable_if](#)< __and_< __is_swappable< _Bilter >, __is_swappable< _Bilter > >::value >::type [swap](#) ([pair](#)< _Bilter, _Bilter > &__x, [pair](#)< _Bilter, _Bilter > &__y) noexcept(noexcept(__x.swap(__y)))
- constexpr bool [operator==](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)
- constexpr bool [operator<](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)
- constexpr bool [operator!=](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)
- constexpr bool [operator>](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)
- constexpr bool [operator<=](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)
- constexpr bool [operator>=](#) (const [pair](#)< _Bilter, _Bilter > &__x, const [pair](#)< _Bilter, _Bilter > &__y)

6.959.1 Detailed Description

template<typename **_Bilter**>
class std::sub_match<_Bilter >

A sequence of characters matched by a particular marked sub-expression.

An object of this class is essentially a pair of iterators marking a matched subexpression within a regular expression pattern match. Such objects can be converted to and compared with std::basic_string objects of a similar base character type as the pattern matched by the regular expression.

The iterators that make up the pair are the usual half-open interval referencing the actual original pattern matched.

6.959.2 Member Typedef Documentation

first_type

typedef _Bilter [std::pair](#)< _Bilter , _Bilter >::first_type [inherited]

The type of the first member.

second_type

typedef _Bilter [std::pair](#)< _Bilter , _Bilter >::second_type [inherited]

The type of the second member.

6.959.3 Member Function Documentation

compare() [1/3]

```
template<typename _BiIter >
int std::sub_match<_BiIter >::compare (
    const string\_type & __s ) const [inline]
```

Compares this sub_match to a string.

Parameters

_ ↔	A string to compare to this sub_match.
_s	

Return values

<i>negative</i>	This matched sequence will collate before __s .
<i>zero</i>	This matched sequence is equivalent to __s .
<i>positive</i>	This matched sequence will collate after __s .

compare() [2/3]

```
template<typename _BiIter >
int std::sub_match< _BiIter >::compare (
    const sub_match< _BiIter > & __s ) const [inline]
```

Compares this and another matched sequence.

Parameters

<code>__s</code>	Another matched sequence to compare to this one.
------------------	--

Return values

<i>negative</i>	This matched sequence will collate before <code>__s</code> .
<i>zero</i>	This matched sequence is equivalent to <code>__s</code> .
<i>positive</i>	This matched sequence will collate after <code>__s</code> .

Referenced by [std::sub_match< _Biliter >::operator!=\(\)](#), [std::sub_match< _Biliter >::operator<\(\)](#), [std::sub_match< _Biliter >::operator<=\(\)](#), [std::sub_match< _Biliter >::operator<=\(\)](#), [std::sub_match< _Biliter >::operator==\(\)](#), [std::sub_match< _Biliter >::operator==\(\)](#), [std::sub_match< _Biliter >::operator==\(\)](#), [std::sub_match< _Biliter >::operator>\(\)](#), and [std::sub_match< _Biliter >::operator>=\(\)](#).

compare() [3/3]

```
template<typename _BiIter >
int std::sub_match< _BiIter >::compare (
    const value_type * __s ) const [inline]
```

Compares this `sub_match` to a string.

Parameters

<code>__s</code>	A string to compare to this <code>sub_match</code> .
------------------	--

Return values

<i>negative</i>	This matched sequence will collate before <code>__s</code> .
<i>zero</i>	This matched sequence is equivalent to <code>__s</code> .
<i>positive</i>	This matched sequence will collate after <code>__s</code> .

length()

```
template<typename _BiIter >
difference_type std::sub_match< _BiIter >::length ( ) const [inline], [noexcept]
```

Gets the length of the matching sequence.

References [std::distance\(\)](#), [std::pair< _Biliter, _Biliter >::first](#), and [std::pair< _Biliter, _Biliter >::second](#).

operator string_type()

```
template<typename _BiIter >
```

```
std::sub_match<_Biliter>::operator string_type ( ) const [inline]
```

Gets the matching sequence as a string.

Returns

the matching sequence as a string.

This is the implicit conversion operator. It is identical to the `str()` member function except that it will want to pop up in unexpected places and cause a great deal of confusion and cursing from the unwary.

References `std::sub_match<_Biliter>::str()`.

`str()`

```
template<typename _Biliter>
```

```
string_type std::sub_match<_Biliter>::str ( ) const [inline]
```

Gets the matching sequence as a string.

Returns

the matching sequence as a string.

References `std::pair<_Biliter, _Biliter>::first`, and `std::pair<_Biliter, _Biliter>::second`.

Referenced by `std::sub_match<_Biliter>::operator string_type()`.

`swap()` [1/2]

```
constexpr void std::pair<_Biliter, _Biliter>::swap (
```

```
pair<_Biliter, _Biliter> & __p ) [inline], [constexpr], [noexcept], [inherited]
```

Swap the first members and then the second members.

`swap()` [2/2]

```
template<typename _Biliter>
```

```
void std::sub_match<_Biliter>::swap (
```

```
sub_match<_Biliter> & __s ) [inline], [noexcept]
```

Swap the values of two `sub_match` objects.

References `std::swap()`, and `std::pair<_T1, _T2>::swap()`.

6.959.4 Friends And Related Symbol Documentation

`make_pair()`

```
constexpr pair<typename __decay_and_strip<_Biliter>::__type, typename __decay_and_strip<_Biliter>::__type> make_pair (
```

```
_Biliter && __x,
```

```
_Biliter && __y ) [related]
```

A convenience wrapper for creating a pair from two objects.

Parameters

<code>__x</code>	The first object.
<code>__y</code>	The second object.

Returns

A newly-constructed `pair<>` object of the appropriate type.

The C++98 standard says the objects are passed by reference-to-const, but C++03 says they are passed by value (this was LWG issue #181).

Since C++11 they have been passed by forwarding reference and then forwarded to the new members of the pair. To create a pair with a member of reference type, pass a `reference_wrapper` to this function.

`operator"!=()`

```
constexpr bool operator!= (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Uses `operator==` to find the result.

`operator<()`

```
constexpr bool operator< (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Defines a lexicographical order for pairs.

For two pairs of the same type, `P` is ordered before `Q` if `P.first` is less than `Q.first`, or if `P.first` and `Q.first` are equivalent (neither is less than the other) and `P.second` is less than `Q.second`.

`operator<=()`

```
constexpr bool operator<= (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Uses `operator<` to find the result.

`operator==(())`

```
constexpr bool operator== (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Two pairs of the same type are equal iff their members are equal.

`operator>()`

```
constexpr bool operator> (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Uses `operator<` to find the result.

`operator>=()`

```
constexpr bool operator>= (
    const pair< _BiIter , _BiIter > & __x,
    const pair< _BiIter , _BiIter > & __y ) [related]
```

Uses `operator<` to find the result.

swap()

```
constexpr enable_if< __and< __is_swappable< _BiIter >, __is_swappable< _BiIter > >::value >↵
::type swap (
    pair< _BiIter , _BiIter > & __x,
    pair< _BiIter , _BiIter > & __y ) [related]
```

Swap overload for pairs. Calls std::pair::swap().

Note

This std::swap overload is not declared in C++03 mode, which has performance implications, e.g. see <https://gcc.gnu.org/PR38466>

6.959.5 Member Data Documentation**first**

```
_BiIter std::pair< _BiIter , _BiIter >::first [inherited]
```

The first member.

second

```
_BiIter std::pair< _BiIter , _BiIter >::second [inherited]
```

The second member.

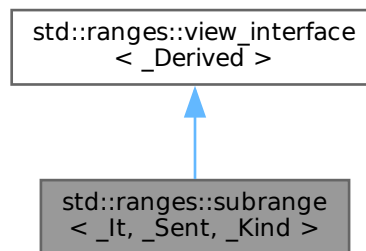
The documentation for this class was generated from the following file:

- [regex.h](#)

6.960 std::ranges::subrange<_It, _Sent, _Kind> Class Template Reference

```
#include <ranges_util.h>
```

Inheritance diagram for std::ranges::subrange<_It, _Sent, _Kind>:

**Public Member Functions**

- constexpr **subrange** (__detail::__convertible_to_non_slicing<_It> auto __i, _Sent __s) noexcept(is_nothrow_constructible_v<_It, decltype(__i)> &&is_nothrow_constructible_v<_Sent, _Sent &>)
- constexpr **subrange** (__detail::__convertible_to_non_slicing<_It> auto __i, _Sent __s, __size_type __n) noexcept(is_nothrow_constructible_v<_It, decltype(__i)> &&is_nothrow_constructible_v<_Sent, _Sent &>)

- `template<__detail::__different_from< subrange > _Rng>`
`requires borrowed_range<_Rng> && __detail::__convertible_to_non_slicing<iterator_t<_Rng>, _It> && convertible_to<sentinel_t<_Rng>, _Sent>`
`constexpr subrange (_Rng &&__r) noexcept(noexcept(subrange(__r, ranges::size(__r)))) _S_store_size`
`&&sized_range<_Rng >`
- `template<__detail::__different_from< subrange > _Rng>`
`requires borrowed_range<_Rng> && __detail::__convertible_to_non_slicing<iterator_t<_Rng>, _It> && convertible_to<sentinel_t<_Rng>, _Sent> (!_S_store_size)`
`constexpr subrange (_Rng &&__r) noexcept(noexcept(subrange(ranges::begin(__r), ranges::end(__r))))`
- `template<borrowed_range _Rng>`
`requires __detail::__convertible_to_non_slicing<iterator_t<_Rng>, _It> && convertible_to<sentinel_t<_Rng>, _Sent> (_Kind ==`
`subrange_kind::sized)`
`constexpr subrange (_Rng &&__r, __size_type __n) noexcept(noexcept(subrange(ranges::begin(__r), ranges::end(__r), __n)))`
- `constexpr subrange & advance (iter_difference_t<_It> __n)`
- `constexpr decltype(auto) back ()`
- `constexpr decltype(auto) back () const`
- `constexpr _It begin ()`
- `constexpr _It begin () const`
- `constexpr auto data () const noexcept(noexcept(ranges::begin(_M_derived())))`
- `constexpr auto data () noexcept(noexcept(ranges::begin(_M_derived())))`
- `constexpr bool empty () const`
- `constexpr bool empty () noexcept(noexcept(_S_empty(_M_derived())))`
- `constexpr _Sent end () const`
- `constexpr decltype(auto) front ()`
- `constexpr decltype(auto) front () const`
- `constexpr subrange next (iter_difference_t<_It> __n=1) &&`
- `constexpr subrange next (iter_difference_t<_It> __n=1) const &`
- `template<__detail::__different_from< subrange > _PairLike>`
`requires __detail::__pair_like_convertible_from<_PairLike, const _It&, const _Sent&>`
`constexpr operator _PairLike () const`
- `constexpr operator bool () const noexcept(noexcept(ranges::empty(_M_derived())))`
- `constexpr operator bool () noexcept(noexcept(ranges::empty(_M_derived())))`
- `template<random_access_range _Range = _Derived>`
`constexpr decltype(auto) operator[] (range_difference_t<_Range> __n)`
- `template<random_access_range _Range = const _Derived>`
`constexpr decltype(auto) operator[] (range_difference_t<_Range> __n) const`
- `constexpr subrange prev (iter_difference_t<_It> __n=1) const`
- `constexpr __size_type size () const`
- `constexpr auto size () noexcept(noexcept(_S_size(_M_derived())))`

Friends

- `struct views::Drop`

6.960.1 Detailed Description

```
template<input_or_output_iterator _It, sentinel_for<_It> _Sent = _It, subrange_kind _Kind = sized_sentinel_for<_Sent, _It> ? subrange_kind::sized : subrange_kind::unsized>
requires (_Kind == subrange_kind::sized || !sized_sentinel_for<_Sent, _It>)
class std::ranges::subrange<_It, _Sent, _Kind>
```

The `ranges::subrange` class template.

The documentation for this class was generated from the following file:

- [ranges_util.h](#)

6.961 std::subtract_with_carry_engine< _UIntType, __w, __s, __r > Class Template Reference

```
#include <random.h>
```

Public Types

- typedef _UIntType [result_type](#)

Public Member Functions

- template<typename _Sseq, typename = _If_seed_seq<_Sseq>>
 [subtract_with_carry_engine](#) (_Sseq &__q)
- [subtract_with_carry_engine](#) ([result_type](#) __sd)
- void [discard](#) (unsigned long long __z)
- [result_type](#) [operator\(\)](#) ()
- template<typename _Sseq >
 _if_seed_seq< _Sseq > [seed](#) (_Sseq &__q)
- template<typename _Sseq >
 auto [seed](#) (_Sseq &__q) -> _If_seed_seq< _Sseq >
- void [seed](#) ([result_type](#) __sd=0u)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr uint_least32_t [default_seed](#)
- static constexpr size_t [long_lag](#)
- static constexpr size_t [short_lag](#)
- static constexpr size_t [word_size](#)

Friends

- template<typename _UIntType1, size_t __w1, size_t __s1, size_t __r1, typename _CharT, typename _Traits >
 std::basic_ostream< _CharT, _Traits > & [operator<<](#) (std::basic_ostream< _CharT, _Traits > &__os, const
 std::subtract_with_carry_engine< _UIntType1, __w1, __s1, __r1 > &__x)
- bool [operator==](#) (const [subtract_with_carry_engine](#) &__lhs, const [subtract_with_carry_engine](#) &__rhs)
- template<typename _UIntType1, size_t __w1, size_t __s1, size_t __r1, typename _CharT, typename _Traits >
 std::basic_istream< _CharT, _Traits > & [operator>>](#) (std::basic_istream< _CharT, _Traits > &__is,
 std::subtract_with_carry_engine< _UIntType1, __w1, __s1, __r1 > &__x)

6.961.1 Detailed Description

```
template<typename _UIntType, size_t __w, size_t __s, size_t __r>
class std::subtract_with_carry_engine< _UIntType, __w, __s, __r >
```

The Marsaglia-Zaman generator.

This is a model of a Generalized Fibonacci discrete random number generator, sometimes referred to as the SWC generator.

A discrete random number generator that produces pseudorandom numbers using:

$$x_i \leftarrow (x_{i-s} - x_{i-r} - carry_{i-1}) \bmod m$$

The size of the state is r and the maximum period of the generator is $(m^r - m^s - 1)$.

6.961.2 Member Typedef Documentation

result_type

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
typedef _UIntType std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::result_type
```

The type of the generated random value.

6.961.3 Constructor & Destructor Documentation

subtract_with_carry_engine() [1/2]

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::subtract_with_carry_engine (
    result_type __sd ) [inline], [explicit]
```

Constructs an explicitly seeded subtract_with_carry_engine random number generator.

References [std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::seed\(\)](#).

subtract_with_carry_engine() [2/2]

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
template<typename _Sseq , typename = _If_seed_seq<_Sseq>>
std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::subtract_with_carry_engine (
    _Sseq & __q ) [inline], [explicit]
```

Constructs a subtract_with_carry_engine random number engine seeded from the seed sequence __q.

Parameters

<code>__q</code>	the seed sequence.
------------------	--------------------

References [std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::seed\(\)](#).

6.961.4 Member Function Documentation

discard()

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
void std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::discard (
    unsigned long long __z ) [inline]
```

Discard a sequence of random numbers.

max()

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
static constexpr result_type std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::max ( )
[inline], [static], [constexpr]
```

Gets the inclusive maximum value of the range of random integers returned by this generator.

min()

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
static constexpr result_type std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::min ( )
[inline], [static], [constexpr]
```

Gets the inclusive minimum value of the range of random integers returned by this generator.

operator()()

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
subtract_with_carry_engine< _UIntType, __w, __s, __r >::result_type std::subtract_with_carry_engine<
_UIntType, __w, __s, __r >::operator()
Gets the next random number in the sequence.
```

seed() [1/2]

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
template<typename _Sseq >
_If_seed_seq< _Sseq > std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::seed (
    _Sseq & __q )
Seeds the initial state  $x_0$  of the % subtract_with_carry_engine random number generator.
```

seed() [2/2]

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
void std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::seed (
    result_type __sd = 0u )
Seeds the initial state  $x_0$  of the random number generator.
```

N1688[4.19] modifies this as follows. If `__value == 0`, sets value to 19780503. In any case, with a linear congruential generator $lcg(i)$ having parameters $m_{lcg} = 2147483563$, $a_{lcg} = 40014$, $c_{lcg} = 0$, and $lcg(0) = value$, sets $x_{-r} \dots x_{-1}$ to $lcg(1) \bmod m \dots lcg(r) \bmod m$ respectively. If $x_{-1} = 0$ set carry to 1, otherwise sets carry to 0.

Referenced by `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::subtract_with_carry_engine()`, and `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::subtract_with_carry_engine()`.

6.961.5 Friends And Related Symbol Documentation**operator<<**

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
template<typename _UIntType1 , size_t __w1, size_t __s1, size_t __r1, typename _CharT , typename
_Traits >
std::basic_ostream< _CharT, _Traits > & operator<< (
    std::basic_ostream< _CharT, _Traits > & __os,
    const std::subtract_with_carry_engine< _UIntType1, __w1, __s1, __r1 > & __x ) [friend]
Inserts the current state of a % subtract_with_carry_engine random number generator engine __x into the output
stream __os.
```

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A % subtract_with_carry_engine random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

operator==

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
bool operator== (
    const subtract_with_carry_engine< _UIntType, __w, __s, __r > & __lhs,
    const subtract_with_carry_engine< _UIntType, __w, __s, __r > & __rhs ) [friend]
```

Compares two % `subtract_with_carry_engine` random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A % <code>subtract_with_carry_engine</code> random number generator object.
<code>__rhs</code>	Another % <code>subtract_with_carry_engine</code> random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

`operator>>`

```
template<typename _UIntType , size_t __w, size_t __s, size_t __r>
template<typename _UIntType1 , size_t __w1, size_t __s1, size_t __r1, typename _CharT , typename
 Traits >
std::basic_istream< _CharT, Traits > & operator>> (
    std::basic_istream< _CharT, Traits > & __is,
    std::subtract_with_carry_engine< _UIntType1, __w1, __s1, __r1 > & __x ) [friend]
```

Extracts the current state of a % `subtract_with_carry_engine` random number generator engine `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A % <code>subtract_with_carry_engine</code> random number generator engine.

Returns

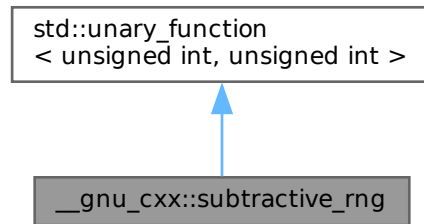
The input stream with the state of `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.962 `__gnu_cxx::subtractive_rng` Class Reference

Inheritance diagram for `__gnu_cxx::subtractive_rng`:



Public Types

- typedef unsigned int [argument_type](#)
- typedef unsigned int [result_type](#)

Public Member Functions

- [subtractive_rng](#) ()
- [subtractive_rng](#) (unsigned int `__seed`)
- void **`_M_initialize`** (unsigned int `__seed`)
- unsigned int [operator\(\)](#) (unsigned int `__limit`)

6.962.1 Detailed Description

The `subtractive_rng` class is documented on [SGI's site](#). Note that this code assumes that `int` is 32 bits.

6.962.2 Member Typedef Documentation

`argument_type`

```
typedef unsigned int std::unary\_function< unsigned int , unsigned int >::argument_type [inherited]
```

`argument_type` is the type of the argument

`result_type`

```
typedef unsigned int std::unary\_function< unsigned int , unsigned int >::result_type [inherited]
```

`result_type` is the return type

6.962.3 Constructor & Destructor Documentation

`subtractive_rng()` [1/2]

```
__gnu_cxx::subtractive_rng::subtractive_rng (
    unsigned int __seed ) [inline]
```

Ctor allowing you to initialize the seed.

subtractive_rng() [2/2]

`__gnu_cxx::subtractive_rng::subtractive_rng () [inline]`
 Default ctor; initializes its state with some number you don't see.

6.962.4 Member Function Documentation**operator>()**

`unsigned int __gnu_cxx::subtractive_rng::operator() (`
 `unsigned int __limit) [inline]`

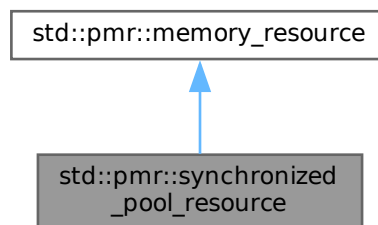
Returns a number less than the argument.

The documentation for this class was generated from the following file:

- [ext/functional](#)

6.963 std::pmr::synchronized_pool_resource Class Reference

Inheritance diagram for `std::pmr::synchronized_pool_resource`:

**Public Member Functions**

- **synchronized_pool_resource** (const [pool_options](#) &__opts)
- **synchronized_pool_resource** (const [pool_options](#) &__opts, [memory_resource](#) *__upstream)
- **synchronized_pool_resource** (const [synchronized_pool_resource](#) &)=delete
- **synchronized_pool_resource** ([memory_resource](#) *__upstream)
- void * **allocate** (size_t __bytes, size_t __alignment=_S_max_align)
- void **deallocate** (void *__p, size_t __bytes, size_t __alignment=_S_max_align)
- bool **is_equal** (const [memory_resource](#) &__other) const noexcept
- **synchronized_pool_resource** & **operator=** (const [synchronized_pool_resource](#) &)=delete
- [pool_options](#) **options** () const noexcept
- void **release** ()
- [memory_resource](#) * **upstream_resource** () const noexcept

Protected Member Functions

- void * **do_allocate** (size_t __bytes, size_t __alignment) override
- void **do_deallocate** (void *__p, size_t __bytes, size_t __alignment) override
- bool **do_is_equal** (const [memory_resource](#) &__other) const noexcept override

6.963.1 Detailed Description

A thread-safe memory resource that manages pools of fixed-size blocks.

6.963.2 Member Function Documentation

`do_allocate()`

```
void * std::pmr::synchronized_pool_resource::do_allocate (
    size_t __bytes,
    size_t __alignment ) [override], [protected], [virtual]
```

Implements [std::pmr::memory_resource](#).

`do_deallocate()`

```
void std::pmr::synchronized_pool_resource::do_deallocate (
    void * __p,
    size_t __bytes,
    size_t __alignment ) [override], [protected], [virtual]
```

Implements [std::pmr::memory_resource](#).

`do_is_equal()`

```
bool std::pmr::synchronized_pool_resource::do_is_equal (
    const memory\_resource & __other ) const [inline], [override], [protected], [virtual],
[noexcept]
```

Implements [std::pmr::memory_resource](#).

The documentation for this class was generated from the following file:

- [memory_resource](#)

6.964 `__gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits >` Struct Template Reference

```
#include <synth_access_traits.hpp>
```

Inherits `_ATraits`.

Public Types

- typedef `_ATraits` **base_type**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `type_traits::const_reference` **const_reference**
- typedef `type_traits::key_const_reference` **key_const_reference**
- typedef `Type_Traits` **type_traits**

Public Member Functions

- **synth_access_traits** (const `base_type` &)
- bool **cmp_keys** (key_const_reference, key_const_reference) const
- bool **cmp_prefixes** (const_iterator, const_iterator, const_iterator, const_iterator, bool compare_after=false) const
- bool **equal_keys** (key_const_reference, key_const_reference) const
- bool **equal_prefixes** (const_iterator, const_iterator, const_iterator, const_iterator, bool compare_after=true) const

Static Public Member Functions

- static key_const_reference **extract_key** (const_reference)

6.964.1 Detailed Description

```
template<typename Type_Traits, bool Set, typename _ATraits>
struct __gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits >
```

Synthetic element access traits.

The documentation for this struct was generated from the following file:

- [synth_access_traits.hpp](#)

6.965 std::chrono::_V2::system_clock Struct Reference

```
#include <chrono.h>
```

Public Types

- typedef [chrono::nanoseconds](#) **duration**
- typedef duration::period **period**
- typedef duration::rep **rep**
- typedef [chrono::time_point](#)< [system_clock](#), [duration](#) > **time_point**

Static Public Member Functions

- static [time_point](#) **from_time_t** (std::time_t __t) noexcept
- static [time_point](#) **now** () noexcept
- static std::time_t **to_time_t** (const [time_point](#) &__t) noexcept

Static Public Attributes

- static constexpr bool **is_steady**

6.965.1 Detailed Description

System clock.

Time returned represents wall time from the system-wide clock.

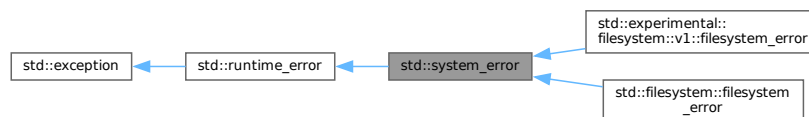
The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.966 std::system_error Class Reference

```
#include <system_error>
```

Inheritance diagram for std::system_error:



Public Member Functions

- **system_error** (const [system_error](#) &)=default
- **system_error** ([error_code](#) __ec, const char *__what)
- **system_error** ([error_code](#) __ec, const [string](#) &__what)
- **system_error** ([error_code](#) __ec=[error_code](#)())
- **system_error** (int __v, const [error_category](#) &__ecat)
- **system_error** (int __v, const [error_category](#) &__ecat, const char *__what)
- **system_error** (int __v, const [error_category](#) &__ecat, const [string](#) &__what)
- const [error_code](#) & **code** () const noexcept
- [system_error](#) & **operator=** (const [system_error](#) &)=default
- virtual const char * **what** () const noexcept

6.966.1 Detailed Description

An exception type that includes an `error_code` value.

Typically used to report errors from the operating system and other low-level APIs.

Since

C++11

6.966.2 Member Function Documentation**what()**

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

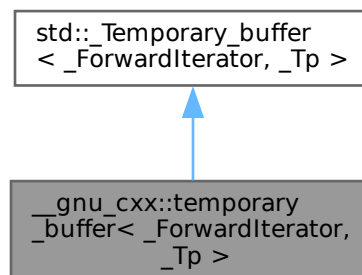
Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

The documentation for this class was generated from the following file:

- [system_error](#)

6.967 `__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>`:



Public Types

- typedef pointer **iterator**
- typedef value_type * **pointer**
- typedef ptrdiff_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- [temporary_buffer](#) (_ForwardIterator __first, _ForwardIterator __last)
- [~temporary_buffer](#) ()
- iterator [begin](#) ()
- iterator [end](#) ()
- size_type [requested_size](#) () const
- size_type [size](#) () const

Protected Attributes

- pointer **_M_buffer**
- size_type **_M_len**
- size_type **_M_original_len**

6.967.1 Detailed Description

```
template<class _ForwardIterator, class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
struct __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >
```

This class provides similar behavior and semantics of the standard functions `get_temporary_buffer()` and `return_temporary_buffer()`, but encapsulated in a type vaguely resembling a standard container.

By default, a `temporary_buffer<Iter>` stores space for objects of whatever type the `Iter` iterator points to. It is constructed from a typical `[first,last)` range, and provides the `begin()`, `end()`, `size()` functions, as well as `requested_size()`. For non-trivial types, copies of `*first` will be used to initialize the storage.

`malloc` is used to obtain underlying storage.

Like `get_temporary_buffer()`, not all the requested memory may be available. Ideally, the created buffer will be large enough to hold a copy of `[first,last)`, but if `size()` is less than `requested_size()`, then this didn't happen.

6.967.2 Constructor & Destructor Documentation

`temporary_buffer()`

```
template<class _ForwardIterator , class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >::temporary_buffer (
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline]
```

Requests storage large enough to hold a copy of `[first,last)`.

`~temporary_buffer()`

```
template<class _ForwardIterator , class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >::~~temporary_buffer ( ) [inline]
```

Destroys objects and frees storage.

6.967.3 Member Function Documentation

`begin()`

```
template<typename _ForwardIterator , typename _Tp >
iterator std::Temporary\_buffer< _ForwardIterator, _Tp >::begin ( ) [inline], [inherited]
```

As per Table mumble.

`end()`

```
template<typename _ForwardIterator , typename _Tp >
iterator std::Temporary\_buffer< _ForwardIterator, _Tp >::end ( ) [inline], [inherited]
```

As per Table mumble.

`requested_size()`

```
template<typename _ForwardIterator , typename _Tp >
size_type std::Temporary\_buffer< _ForwardIterator, _Tp >::requested_size ( ) const [inline],
[inherited]
```

Returns the size requested by the constructor; may be `>size()`.

`size()`

```
template<typename _ForwardIterator , typename _Tp >
size_type std::Temporary\_buffer< _ForwardIterator, _Tp >::size ( ) const [inline], [inherited]
```

As per Table mumble.

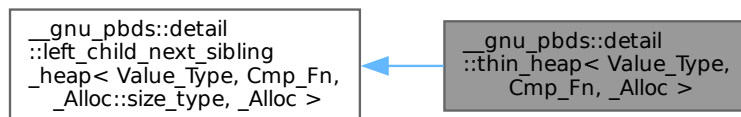
The documentation for this struct was generated from the following file:

- [ext/memory](#)

6.968 `__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

```
#include <thin_heap.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_a::const_pointer` **const_pointer**
- typedef `__rebind_a::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**

- typedef [base_type::iterator](#) **iterator**
- typedef [base_type::point_const_iterator](#) **point_const_iterator**
- typedef [base_type::point_iterator](#) **point_iterator**
- typedef [__rebind_a::pointer](#) **pointer**
- typedef [__rebind_a::reference](#) **reference**
- typedef [_Alloc::size_type](#) **size_type**
- typedef [Value_Type](#) **value_type**

Public Member Functions

- [iterator](#) **begin** ()
- [const_iterator](#) **begin** () const
- void **clear** ()
- bool **empty** () const
- [iterator](#) **end** ()
- [const_iterator](#) **end** () const
- void **erase** ([point_iterator](#))
- template<typename Pred >
size_type **erase_if** (Pred)
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- void **join** ([thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- size_type **max_size** () const
- void **modify** ([point_iterator](#), const_reference)
- void **pop** ()
- [point_iterator](#) **push** (const_reference)
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, [thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap** ([left_child_next_sibling_heap](#)< Value_Type, Cmp_Fn, _Alloc::size_type, _Alloc > &)
- const_reference **top** () const

Protected Types

- typedef [base_type::node](#) **node**
- typedef [alloc_traits::allocator_type](#) **node_allocator**
- typedef [base_type::node_const_pointer](#) **node_const_pointer**
- typedef [_Alloc::size_type](#) **node_metadata**
- typedef [base_type::node_pointer](#) **node_pointer**
- typedef [std::pair](#)< node_pointer, node_pointer > **node_pointer_pair**

Protected Member Functions

- **thin_heap** (const Cmp_Fn &)
- **thin_heap** (const [thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)

- void **swap** ([thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** ([left_child_next_sibling_heap](#) &)

Static Protected Member Functions

- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_root**
- size_type **m_size**

6.968.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >
```

Thin heap.

See Tarjan and Kaplan.

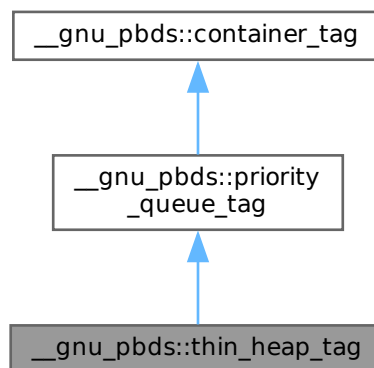
The documentation for this class was generated from the following file:

- [thin_heap.hpp](#)

6.969 __gnu_pbds::thin_heap_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::thin_heap_tag:



6.969.1 Detailed Description

Thin heap.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.970 std::thread Class Reference

```
#include <thread>
```

Classes

- class [id](#)

Public Types

- using [_State_ptr](#) = [unique_ptr](#)< [_State](#) >
- using [native_handle_type](#) = [__gthread_t](#)

Public Member Functions

- `template<typename _Callable, typename... _Args, typename = _Require<__not_same<_Callable>>>>`
`thread (_Callable &&__f, _Args &&... __args)`
- `thread (const thread &)=delete`
- `thread (thread &&__t) noexcept`
- `void detach ()`
- `id get_id () const noexcept`
- `void join ()`
- `bool joinable () const noexcept`
- `native_handle_type native_handle ()`
- `thread & operator= (const thread &)=delete`
- `thread & operator= (thread &&__t) noexcept`
- `void swap (thread &__t) noexcept`

Static Public Member Functions

- static unsigned int [hardware_concurrency](#) () noexcept

Related Symbols

(Note that these are not member symbols.)

- void [swap](#) ([thread](#) &__x, [thread](#) &__y) noexcept

6.970.1 Detailed Description

A `std::thread` represents a new thread of execution.

The default constructor creates an object that does not own a thread. The `thread(F&&, Args&&...)` constructor invokes a callable in a new thread, and owns that new thread. A `std::thread` that owns a thread is *joinable*. Joining a thread waits for it to finish executing, which happens when the callable running in that thread returns.

A `std::thread` cannot be copied, but can be moved. Moving a joinable object transfers ownership of its thread to another object.

A joinable `std::thread` must be explicitly joined (or detached) before it is destroyed or assigned to. Attempting to destroy a joinable thread will terminate the whole process.

Since

C++11

6.970.2 Member Function Documentation

`native_handle()`

```
native_handle_type std::thread::native_handle ( ) [inline]
```

Precondition

thread is joinable

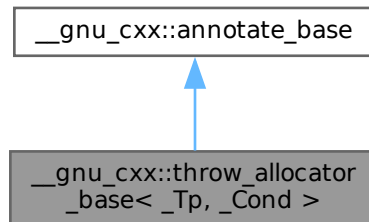
The documentation for this class was generated from the following file:

- [std_thread.h](#)

6.971 `__gnu_cxx::throw_allocator_base<_Tp, _Cond>` Class Template Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for `__gnu_cxx::throw_allocator_base<_Tp, _Cond>`:



Public Types

- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef std::ptrdiff_t **difference_type**
- typedef value_type * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef value_type & **reference**
- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- const_pointer **address** (const_reference __x) const noexcept
- pointer **address** (reference __x) const noexcept
- pointer **allocate** (size_type __n, const void *__hint=0)
- void **check** (size_t label)
- void **check** (size_type __n)
- void **check_allocated** (pointer __p, size_type __n)
- map_alloc_type::iterator **check_allocated** (void *p, size_t size)
- void **check_constructed** (size_t label)

- `map_construct_type::iterator` **check_constructed** (void *p)
- `template<typename _Up, typename... _Args>`
void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- `template<typename _Up >`
void **destroy** (_Up *__p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)
- size_type **max_size** () const noexcept

Static Public Member Functions

- static void **check** ()
- static size_t **get_label** ()
- static void **set_label** (size_t l)

6.971.1 Detailed Description

```
template<typename _Tp, typename _Cond>
class __gnu_cxx::throw_allocator_base< _Tp, _Cond >
```

Allocator class with logging and exception generation control. Intended to be used as an `allocator_type` in templated code.

Note: Deallocate not allowed to throw.

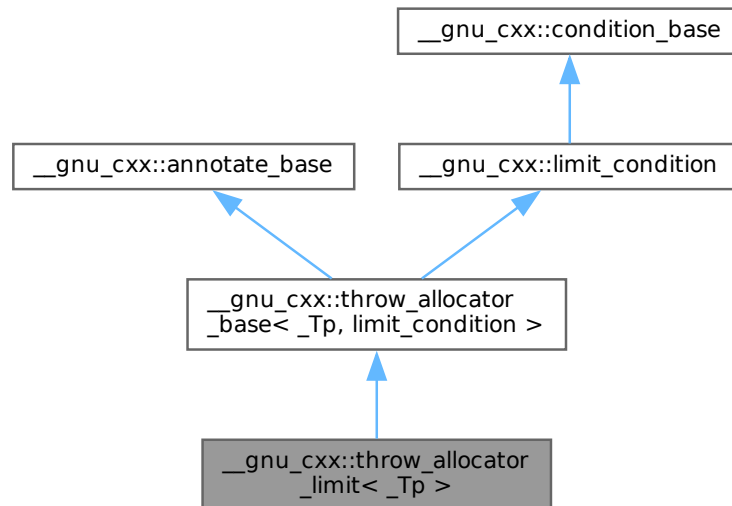
The documentation for this class was generated from the following file:

- [throw_allocator.h](#)

6.972 __gnu_cxx::throw_allocator_limit< _Tp > Struct Template Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for `__gnu_cxx::throw_allocator_limit<_Tp>`:



Public Types

- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef std::ptrdiff_t **difference_type**
- typedef value_type * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef value_type & **reference**
- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **throw_allocator_limit** (const [throw_allocator_limit](#) &) noexcept
- template<typename _Tp1 >
 throw_allocator_limit (const [throw_allocator_limit](#)<_Tp1 > &) noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **address** (reference __x) const noexcept
- pointer **allocate** (size_type __n, const void * __hint=0)
- void **check** (size_t label)
- void **check** (size_type __n)
- void **check_allocated** (pointer __p, size_type __n)
- map_alloc_type::iterator **check_allocated** (void *p, size_t size)
- void **check_constructed** (size_t label)
- map_construct_type::iterator **check_constructed** (void *p)
- void **construct** (_Up * __p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)

- void **destroy** (_Up *__p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)
- size_type **max_size** () const noexcept
- **throw_allocator_limit** & **operator=** (const **throw_allocator_limit** &)=default

Static Public Member Functions

- static void **check** ()
- static size_t & **count** ()
- static size_t **get_label** ()
- static size_t & **limit** ()
- static void **set_label** (size_t l)
- static void **set_limit** (const size_t __l)
- static void **throw_conditionally** ()

6.972.1 Detailed Description

```
template<typename _Tp>
struct __gnu_cxx::throw_allocator_limit< _Tp >
```

Allocator throwing via limit condition.

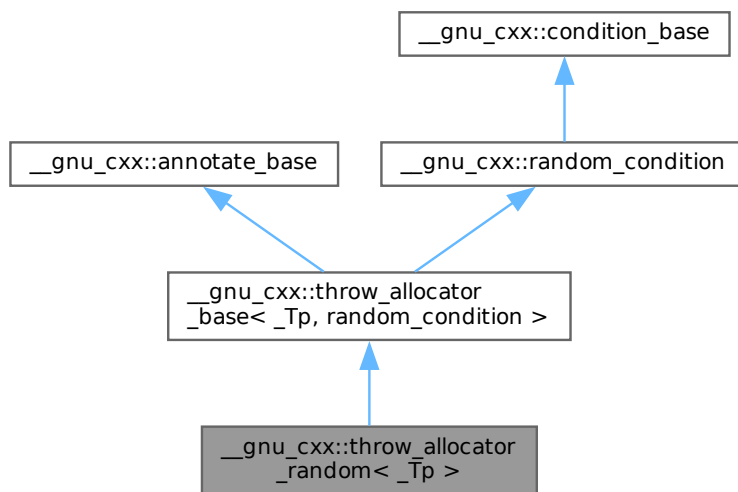
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.973 __gnu_cxx::throw_allocator_random< _Tp > Struct Template Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for __gnu_cxx::throw_allocator_random< _Tp >:



Public Types

- `typedef const value_type * const_pointer`
- `typedef const value_type & const_reference`
- `typedef std::ptrdiff_t difference_type`
- `typedef value_type * pointer`
- `typedef std::true_type propagate_on_container_move_assignment`
- `typedef value_type & reference`
- `typedef std::size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `throw_allocator_random` (const [throw_allocator_random](#) &) noexcept
- `template<typename _Tp1 >`
`throw_allocator_random` (const [throw_allocator_random](#)<_Tp1> &) noexcept
- `const_pointer address` (const_reference __x) const noexcept
- `pointer address` (reference __x) const noexcept
- `pointer allocate` (size_type __n, const void * __hint=0)
- `void check` (size_t label)
- `void check` (size_type __n)
- `void check_allocated` (pointer __p, size_type __n)
- `map_alloc_type::iterator check_allocated` (void *p, size_t size)
- `void check_constructed` (size_t label)
- `map_construct_type::iterator check_constructed` (void *p)
- `void construct` (_Up * __p, _Args &&... __args)
- `void deallocate` (pointer __p, size_type __n)
- `void destroy` (_Up * __p)
- `void erase` (void *p, size_t size)
- `void erase_construct` (void *p)
- `void insert` (void *p, size_t size)
- `void insert_construct` (void *p)
- `size_type max_size` () const noexcept
- `throw_allocator_random & operator=` (const [throw_allocator_random](#) &)=default
- `void seed` (unsigned long __s)

Static Public Member Functions

- `static void check` ()
- `static size_t get_label` ()
- `static void set_label` (size_t l)
- `static void set_probability` (double __p)
- `static void throw_conditionally` ()

6.973.1 Detailed Description

```
template<typename _Tp>
struct __gnu_cxx::throw_allocator_random<_Tp>
```

Allocator throwing via random condition.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.974 `__gnu_cxx::throw_value_base<_Cond>` Struct Template Reference

```
#include <throw_allocator.h>  
Inherits _Cond.
```

Public Types

- typedef `_Cond` **`condition_type`**

Public Member Functions

- **`throw_value_base`** (const std::size_t __i)
- **`throw_value_base`** (const [throw_value_base](#) &__v)
- **`throw_value_base`** ([throw_value_base](#) &&)=default
- [throw_value_base](#) & **`operator++`** ()
- [throw_value_base](#) & **`operator=`** (const [throw_value_base](#) &__v)
- [throw_value_base](#) & **`operator=`** ([throw_value_base](#) &&)=default

Public Attributes

- std::size_t **`_M_i`**

6.974.1 Detailed Description

```
template<typename _Cond>  
struct __gnu_cxx::throw_value_base<_Cond>
```

Class with exception generation control. Intended to be used as a `value_type` in templated code.

Note: Destructor not allowed to throw.

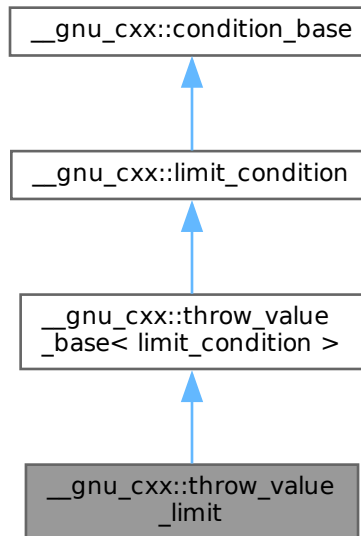
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.975 `__gnu_cxx::throw_value_limit` Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for `__gnu_cxx::throw_value_limit`:



Public Types

- typedef `throw_value_base< limit_condition >` `base_type`
- typedef `limit_condition` `condition_type`

Public Member Functions

- `throw_value_limit` (const std::size_t __i)
- `throw_value_limit` (const `throw_value_limit` &__other)
- `throw_value_limit` (`throw_value_limit` &&)=default
- `throw_value_base` & `operator++` ()
- `throw_value_limit` & `operator=` (const `throw_value_limit` &__other)
- `throw_value_limit` & `operator=` (`throw_value_limit` &&)=default

Static Public Member Functions

- static size_t & `count` ()
- static size_t & `limit` ()
- static void `set_limit` (const size_t __l)
- static void `throw_conditionally` ()

Public Attributes

- std::size_t `M_i`

6.975.1 Detailed Description

Type throwing via limit condition.

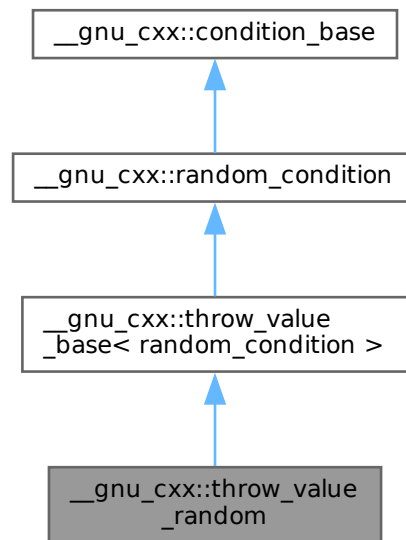
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.976 __gnu_cxx::throw_value_random Struct Reference

```
#include <throw_allocator.h>
```

Inheritance diagram for __gnu_cxx::throw_value_random:



Public Types

- typedef [throw_value_base< random_condition >](#) **base_type**
- typedef [random_condition](#) **condition_type**

Public Member Functions

- **throw_value_random** (const std::size_t __i)
- **throw_value_random** (const [throw_value_random](#) &__other)
- **throw_value_random** ([throw_value_random](#) &&)=default
- [throw_value_base](#) & **operator++** ()
- [throw_value_random](#) & **operator=** (const [throw_value_random](#) &__other)
- [throw_value_random](#) & **operator=** ([throw_value_random](#) &&)=default
- void **seed** (unsigned long __s)

Static Public Member Functions

- static void **set_probability** (double __p)
- static void **throw_conditionally** ()

Public Attributes

- `std::size_t _M_i`

6.976.1 Detailed Description

Type throwing via random condition.

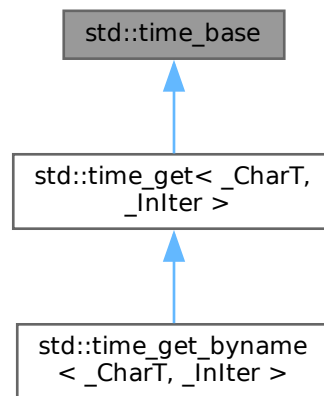
The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

6.977 std::time_base Class Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::time_base`:

**Public Types**

- enum `dateorder` {
`no_order`, `dmy`, `mdy`, `ymd`,
`ydm` }

6.977.1 Detailed Description

Time format ordering data.

This class provides an enum representing different orderings of time: day, month, and year.

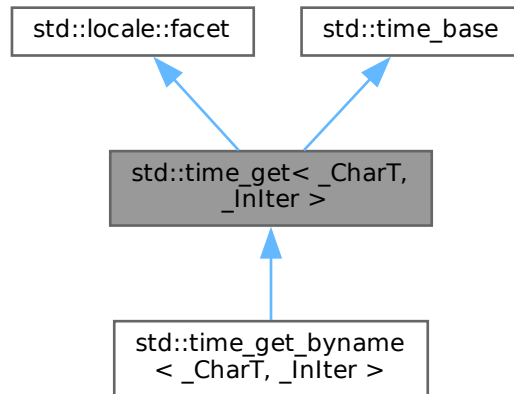
The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

6.978 std::time_get<_CharT, _Inlter> Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::time_get<_CharT, _InIter>`:



Public Types

- enum `dateorder` {
 `no_order` , `dmy` , `mdy` , `ymd` ,
 `ydm` }
- typedef `_CharT` `char_type`
- typedef `_InIter` `iter_type`

Public Member Functions

- `time_get` (`size_t` __refs=0)
- `dateorder` `date_order` () const
- `iter_type` `get` (`iter_type` __s, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm, `char` __format, `char` __modifier=0) const
- `iter_type` `get` (`iter_type` __s, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm, const `char_type` *__fmt, const `char_type` *__fmtend) const
- `iter_type` `get_date` (`iter_type` __beg, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm) const
- `iter_type` `get_monthname` (`iter_type` __beg, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm) const
- `iter_type` `get_time` (`iter_type` __beg, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm) const
- `iter_type` `get_weekday` (`iter_type` __beg, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm) const
- `iter_type` `get_year` (`iter_type` __beg, `iter_type` __end, `ios_base` &__io, `ios_base::iostate` &__err, `tm` *__tm) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~time_get()`
- `iter_type M_extract_name(iter_type __beg, iter_type __end, int &__member, const _CharT **__names, size_t __indexlen, ios_base &__io, ios_base::iostate &__err) const`
- `iter_type M_extract_num(iter_type __beg, iter_type __end, int &__member, int __min, int __max, size_t __len, ios_base &__io, ios_base::iostate &__err) const`
- `iter_type M_extract_via_format(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm, const _CharT *__format) const`
- `iter_type M_extract_via_format(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm, const _CharT *__format, __time_get_state &__state) const`
- `iter_type M_extract_wday_or_month(iter_type __beg, iter_type __end, int &__member, const _CharT **__names, size_t __indexlen, ios_base &__io, ios_base::iostate &__err) const`
- virtual dateorder `do_date_order()` const
- `iter_type do_get(iter_type __s, iter_type __end, ios_base &__f, ios_base::iostate &__err, tm *__tm, char __format, char __modifier) const`
- virtual `iter_type do_get_date(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_monthname(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_time(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_weekday(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_year(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale &__cloc) throw()`
- static void `_S_create_c_locale(__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale(__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale()`
- static const char * `_S_get_c_name()` throw()
- static `__c_locale _S_lc_type_c_locale(__c_locale __cloc, const char *__s)`

6.978.1 Detailed Description

template<typename _CharT, typename _InIter>

class std::time_get<_CharT, _InIter>

Primary class template `time_get`.

This facet encapsulates the code to parse and return a date or time from a string. It is used by the istream numeric extraction operators.

The `time_get` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `time_get` facet.

6.978.2 Member Typedef Documentation

char_type

```
template<typename _CharT, typename _InIter>
```

```
typedef _CharT std::time_get<_CharT, _InIter>::char_type
```

Public typedefs.

iter_type

```
template<typename _CharT , typename _InIter >
typedef _InIter std::time_get< _CharT, _InIter >::iter_type
Public typedefs.
```

6.978.3 Constructor & Destructor Documentation

time_get()

```
template<typename _CharT , typename _InIter >
std::time_get< _CharT, _InIter >::time_get (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~time_get()

```
template<typename _CharT , typename _InIter >
virtual std::time_get< _CharT, _InIter >::~~time_get ( ) [inline], [protected], [virtual]
Destructor.
```

6.978.4 Member Function Documentation

date_order()

```
template<typename _CharT , typename _InIter >
dateorder std::time_get< _CharT, _InIter >::date_order ( ) const [inline]
```

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_↵put::put()` only uses month, day, and year. If the format `x` for the associated locale uses other fields, this function returns `time_base::dateorder::noorder`.

NOTE: The library always returns `noorder` at the moment.

Returns

A member of `time_base::dateorder`.

References `std::time_get< _CharT, _InIter >::do_date_order()`.

do_date_order()

```
template<typename _CharT , typename _InIter >
_GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11 time_base::dateorder std::time_get< _CharT, _InIter >::do_↵
date_order [protected], [virtual]
```

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_↵put::put()` only uses month, day, and year. This function is a hook for derived classes to change the value returned.

Returns

A member of `time_base::dateorder`.

Referenced by `std::time_get< _CharT, _InIter >::date_order()`.

do_get()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get (
    iter_type __s,
    iter_type __end,
    ios_base & __f,
    ios_base::iostate & __err,
    tm * __tm,
    char __format,
    char __modifier ) const [inline], [protected]
```

Parse input string according to format.

This function parses the string according to the provided format and optional modifier. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__f</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

References [std::ios_base::M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), and [std::__ctype_abstract_base< _CharT >::widen\(\)](#).
Referenced by [std::time_get< _CharT, _InIter >::get\(\)](#), and [std::time_get< _CharT, _InIter >::get\(\)](#).

do_get_date()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_date (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual]
```

Parse input date string.

This function parses a date according to the format *X* and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

[get_date\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

References [std::ios_base::_M_getloc\(\)](#), and [std::ios_base::eofbit](#).

Referenced by [std::time_get<_CharT, _InIter>::get_date\(\)](#).

do_get_monthname()

```
template<typename _CharT, typename _InIter>
_InIter std::time_get<_CharT, _InIter>::do_get_monthname (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

[get_monthname\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

References [std::ios_base::_M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), and [std::ios_base::goodbit](#).

Referenced by [std::time_get<_CharT, _InIter>::get_monthname\(\)](#).

do_get_time()

```
template<typename _CharT, typename _InIter>
_InIter std::time_get<_CharT, _InIter>::do_get_time (
    iter_type __beg,
    iter_type __end,
```

```

ios_base & __io,
ios_base::iostate & __err,
tm * __tm ) const [protected], [virtual]

```

Parse input time string.

This function parses a time according to the format *x* and puts the results into a user-supplied struct *tm*. This function is a hook for derived classes to change the value returned.

See also

[get_time\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond time string.

References [std::ios_base::_M_getloc\(\)](#), and [std::ios_base::eofbit](#).

Referenced by [std::time_get< _CharT, _InIter >::get_time\(\)](#).

do_get_weekday()

```

template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_weekday (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual]

```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct *tm*. This function is a hook for derived classes to change the value returned.

See also

[get_weekday\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond weekday name.

References [std::ios_base::M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), and [std::ios_base::goodbit](#).

Referenced by [std::time_get<_CharT, _InIter>::get_weekday\(\)](#).

do_get_year()

```
template<typename _CharT, typename _InIter >
_InIter std::time_get<_CharT, _InIter>::do_get_year (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

[get_year\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

References [std::ios_base::M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), and [std::ctype_abstract_base<_CharT>::narrow\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get_year\(\)](#).

get() [1/2]

```
template<typename _CharT, typename _InIter >
iter_type std::time_get<_CharT, _InIter>::get (
    iter_type __s,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm,
    char __format,
    char __modifier = 0 ) const [inline]
```

Parse input string according to format.

This function calls [time_get::do_get](#) with the provided parameters.

See also

do_get() and get().

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

References [std::time_get< _CharT, _InIter >::do_get\(\)](#).

get() [2/2]

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::get (
    iter_type __s,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm,
    const char_type * __fmt,
    const char_type * __fmtend ) const [inline]
```

Parse input string according to format.

This function parses the input string according to a provided format string. It does the inverse of `time_put::put`. The format string follows the format specified for `strptime(3)/strptime(3)`. The actual parsing is done by `time_get::do_get`.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__fmt</code>	Start of the format string.
<code>__fmtend</code>	End of the format string.

Returns

Iterator to first char not parsed.

References [std::ios_base::_M_getloc\(\)](#), [std::time_get< _CharT, _InIter >::do_get\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::__ctype_abstract_base< _CharT >::is\(\)](#), [std::__ctype_abstract_base< _CharT >::narrow\(\)](#), [std::__ctype_abstract_base< _CharT >::tolower\(\)](#), and [std::__ctype_abstract_base< _CharT >::toupper\(\)](#).

get_date()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_date (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline]
```

Parse input date string.

This function parses a date according to the format *x* and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_date()`.

If there is a valid date string according to format *x*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the date string. If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond date string.

References `std::time_get< _CharT, _InIter >::do_get_date()`.

get_monthname()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_monthname (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_monthname()`.

Parsing starts by parsing an abbreviated month name. If a valid abbreviation is followed by a character that would lead to the full month name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond month name.

References [std::time_get< _CharT, _InIter >::do_get_monthname\(\)](#).

get_time()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_time (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline]
```

Parse input time string.

This function parses a time according to the format *X* and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_time()`.

If there is a valid time string according to format *X*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the time string. If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond time string.

References [std::time_get< _CharT, _InIter >::do_get_time\(\)](#).

get_weekday()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_weekday (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_weekday()`.

Parsing starts by parsing an abbreviated weekday name. If a valid abbreviation is followed by a character that would lead to the full weekday name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond weekday name.

References [std::time_get<_CharT, _InIter>::do_get_weekday\(\)](#).

get_year()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_year (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_year()`.

4 consecutive digits are interpreted as a full year. If there are exactly 2 consecutive digits, the library interprets this as the number of years since 1900.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

References [std::time_get<_CharT, _InIter>::do_get_year\(\)](#).

6.978.5 Member Data Documentation**id**

```
template<typename _CharT , typename _InIter >
locale::id std::time_get< _CharT, _InIter >::id [static]
```

Numpunct facet id.

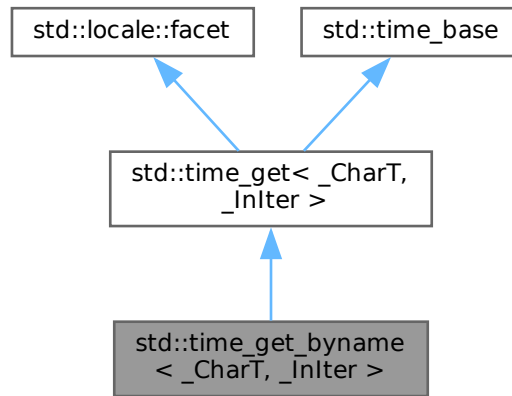
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

6.979 std::time_get_byname< _CharT, _InIter > Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::time_get_byname< _CharT, _InIter >:

**Public Types**

- typedef `_CharT` **char_type**
- enum **dateorder** {
 no_order , **dmy** , **mdy** , **ymd** ,
 ydm }
- typedef `_InIter` **iter_type**

Public Member Functions

- **time_get_byname** (const char *, size_t __refs=0)
- **time_get_byname** (const [string](#) &__s, size_t __refs=0)
- dateorder [date_order](#) () const
- iter_type [get](#) (iter_type __s, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm, char __format, char __modifier=0) const
- iter_type [get](#) (iter_type __s, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm, const char_type *__fmt, const char_type *__fmtend) const
- iter_type [get_date](#) (iter_type __beg, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm) const
- iter_type [get_monthname](#) (iter_type __beg, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm) const
- iter_type [get_time](#) (iter_type __beg, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm) const
- iter_type [get_weekday](#) (iter_type __beg, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm) const
- iter_type [get_year](#) (iter_type __beg, iter_type __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, tm *__tm) const

Static Public Attributes

- static [locale::id](#) **id**

Protected Member Functions

- `iter_type _M_extract_name` (`iter_type __beg`, `iter_type __end`, `int &__member`, `const _CharT **__names`, `size_t __indexlen`, `ios_base &__io`, `ios_base::iostate &__err`) `const`
- `iter_type _M_extract_num` (`iter_type __beg`, `iter_type __end`, `int &__member`, `int __min`, `int __max`, `size_t __len`, `ios_base &__io`, `ios_base::iostate &__err`) `const`
- `iter_type _M_extract_via_format` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `tm *__tm`, `const _CharT *__format`) `const`
- `iter_type _M_extract_via_format` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `tm *__tm`, `const _CharT *__format`, `__time_get_state &__state`) `const`
- `iter_type _M_extract_wday_or_month` (`iter_type __beg`, `iter_type __end`, `int &__member`, `const _CharT **__names`, `size_t __indexlen`, `ios_base &__io`, `ios_base::iostate &__err`) `const`
- virtual `dateorder do_date_order` () `const`
- `iter_type do_get` (`iter_type __s`, `iter_type __end`, `ios_base &__f`, `ios_base::iostate &__err`, `tm *__tm`, `char __format`, `char __modifier`) `const`
- virtual `iter_type do_get_date` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `tm *__tm`) `const`
- virtual `iter_type do_get_monthname` (`iter_type __beg`, `iter_type __end`, `ios_base &`, `ios_base::iostate &__err`, `tm *__tm`) `const`
- virtual `iter_type do_get_time` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `tm *__tm`) `const`
- virtual `iter_type do_get_weekday` (`iter_type __beg`, `iter_type __end`, `ios_base &`, `ios_base::iostate &__err`, `tm *__tm`) `const`
- virtual `iter_type do_get_year` (`iter_type __beg`, `iter_type __end`, `ios_base &__io`, `ios_base::iostate &__err`, `tm *__tm`) `const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale` (`__c_locale &__cloc`) `throw ()`
- static `void _S_create_c_locale` (`__c_locale &__cloc`, `const char *__s`, `__c_locale __old=0`)
- static `void _S_destroy_c_locale` (`__c_locale &__cloc`)
- static `__c_locale _S_get_c_locale` ()
- static `const char * _S_get_c_name` () `throw ()`
- static `__c_locale _S_lc_ctype_c_locale` (`__c_locale __cloc`, `const char *__s`)

6.979.1 Detailed Description

```
template<typename _CharT, typename _InIter>
```

```
class std::time_get_byname< _CharT, _InIter >
```

class `time_get_byname` [22.2.5.2].

6.979.2 Member Function Documentation

`date_order()`

```
template<typename _CharT , typename _InIter >
```

```
dateorder std::time_get< _CharT, _InIter >::date_order ( ) const [inline], [inherited]
```

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_put::put()` only uses month, day, and year. If the format `x` for the associated locale uses other fields, this function returns `time_base::dateorder::noorder`.

NOTE: The library always returns `noorder` at the moment.

Returns

A member of time_base::dateorder.

References [std::time_get<_CharT, _InIter>::do_date_order\(\)](#).

do_date_order()

```
template<typename _CharT, typename _InIter>
_GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11 time_base::dateorder std::time_get<_CharT, _InIter>::do_date_order [protected], [virtual], [inherited]
```

Return preferred order of month, day, and year.

This function returns an enum from time_base::dateorder giving the preferred ordering if the format x given to time_get::put() only uses month, day, and year. This function is a hook for derived classes to change the value returned.

Returns

A member of time_base::dateorder.

Referenced by [std::time_get<_CharT, _InIter>::date_order\(\)](#).

do_get()

```
template<typename _CharT, typename _InIter>
_InIter std::time_get<_CharT, _InIter>::do_get (
    iter_type __s,
    iter_type __end,
    ios_base & __f,
    ios_base::iostate & __err,
    tm * __tm,
    char __format,
    char __modifier) const [inline], [protected], [inherited]
```

Parse input string according to format.

This function parses the string according to the provided format and optional modifier. This function is a hook for derived classes to change the value returned.

See also

[get\(\)](#) for more details.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__f</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

References [std::ios_base::M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::goodbit](#), and [std::__ctype_abstract_base<_CharT>::widen\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get\(\)](#), and [std::time_get<_CharT, _InIter>::get\(\)](#).

do_get_date()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_date (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual], [inherited]
```

Parse input date string.

This function parses a date according to the format *X* and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_date()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

References `std::ios_base::_M_getloc()`, and `std::ios_base::eofbit`.

Referenced by `std::time_get< _CharT, _InIter >::get_date()`.

do_get_monthname()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_monthname (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual], [inherited]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_monthname()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

References [std::ios_base::_M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), and [std::ios_base::goodbit](#).

Referenced by [std::time_get< _CharT, _InIter >::get_monthname\(\)](#).

do_get_time()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_time (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual], [inherited]
```

Parse input time string.

This function parses a time according to the format *x* and puts the results into a user-supplied struct *tm*. This function is a hook for derived classes to change the value returned.

See also

[get_time\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond time string.

References [std::ios_base::_M_getloc\(\)](#), and [std::ios_base::eofbit](#).

Referenced by [std::time_get< _CharT, _InIter >::get_time\(\)](#).

do_get_weekday()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_weekday (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual], [inherited]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct *tm*. This function is a hook for derived classes to change the value returned.

See also

[get_weekday\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond weekday name.

References [std::ios_base::_M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), and [std::ios_base::goodbit](#).

Referenced by [std::time_get<_CharT, _InIter>::get_weekday\(\)](#).

do_get_year()

```
template<typename _CharT , typename _InIter >
_InIter std::time_get< _CharT, _InIter >::do_get_year (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [protected], [virtual], [inherited]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

[get_year\(\)](#) for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

References [std::ios_base::_M_getloc\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), and [std::_ctype_abstract_base<_CharT>::narrow\(\)](#).

Referenced by [std::time_get<_CharT, _InIter>::get_year\(\)](#).

get() [1/2]

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get (
    iter_type __s,
```

```

    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm,
    char __format,
    char __modifier = 0 ) const [inline], [inherited]

```

Parse input string according to format.

This function calls time_get::do_get with the provided parameters.

See also

do_get() and get().

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

References [std::time_get<_CharT, _InIter >::do_get\(\)](#).

get() [2/2]

```

template<typename _CharT , typename _InIter >
_InIter std::time_get<_CharT, _InIter >::get (
    iter_type __s,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm,
    const char_type * __fmt,
    const char_type * __fmtend ) const [inline], [inherited]

```

Parse input string according to format.

This function parses the input string according to a provided format string. It does the inverse of time_put::put. The format string follows the format specified for strftime(3)/strptime(3). The actual parsing is done by time_get::do_get.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__fmt</code>	Start of the format string.
<code>__fmtend</code>	End of the format string.

Returns

Iterator to first char not parsed.

References [std::ios_base::_M_getloc\(\)](#), [std::time_get<_CharT, _InIter>::do_get\(\)](#), [std::ios_base::eofbit](#), [std::ios_base::failbit](#), [std::ios_base::goodbit](#), [std::__ctype_abstract_base<_CharT>::is\(\)](#), [std::__ctype_abstract_base<_CharT>::narrow\(\)](#), [std::__ctype_abstract_base<_CharT>::tolower\(\)](#), and [std::__ctype_abstract_base<_CharT>::toupper\(\)](#).

get_date()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get<_CharT, _InIter>::get_date (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline], [inherited]
```

Parse input date string.

This function parses a date according to the format *x* and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_date()`.

If there is a valid date string according to format *x*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the date string. If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond date string.

References [std::time_get<_CharT, _InIter>::do_get_date\(\)](#).

get_monthname()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get<_CharT, _InIter>::get_monthname (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline], [inherited]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_monthname()`.

Parsing starts by parsing an abbreviated month name. If a valid abbreviation is followed by a character that would lead to the full month name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

References [std::time_get<_CharT, _InIter >::do_get_monthname\(\)](#).

get_time()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get<_CharT, _InIter >::get_time (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline], [inherited]
```

Parse input time string.

This function parses a time according to the format *X* and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_time()`.

If there is a valid time string according to format *X*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the time string. If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond time string.

References [std::time_get<_CharT, _InIter >::do_get_time\(\)](#).

get_weekday()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get<_CharT, _InIter >::get_weekday (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline], [inherited]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_weekday()`.

Parsing starts by parsing an abbreviated weekday name. If a valid abbreviation is followed by a character that would lead to the full weekday name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond weekday name.

References [std::time_get<_CharT, _InIter>::do_get_weekday\(\)](#).

get_year()

```
template<typename _CharT , typename _InIter >
iter_type std::time_get< _CharT, _InIter >::get_year (
    iter_type __beg,
    iter_type __end,
    ios_base & __io,
    ios_base::iostate & __err,
    tm * __tm ) const [inline], [inherited]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_year()`.

4 consecutive digits are interpreted as a full year. If there are exactly 2 consecutive digits, the library interprets this as the number of years since 1900.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

References `std::time_get< _CharT, _InIter >::do_get_year()`.

6.979.3 Member Data Documentation

id

template<typename _CharT , typename _InIter >
`locale::id std::time_get< _CharT, _InIter >::id` [static], [inherited]
Numpunct facet id.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

6.980 `std::chrono::time_point< _Clock, _Dur >` Class Template Reference

Public Types

- typedef `_Clock` **clock**
- typedef `_Dur` **duration**
- typedef `duration::period` **period**
- typedef `duration::rep` **rep**

Public Member Functions

- constexpr **time_point** (const duration &__dur)
- template<typename _Dur2 , typename = _Require<is_convertible<_Dur2, _Dur>>>
constexpr **time_point** (const [time_point](#)< clock, _Dur2 > &__t)
- constexpr [time_point](#) & **operator++** ()
- constexpr [time_point](#) **operator++** (int)
- constexpr [time_point](#) & **operator+=** (const duration &__dur)
- constexpr [time_point](#) & **operator--** ()
- constexpr [time_point](#) **operator--** (int)
- constexpr [time_point](#) & **operator-=** (const duration &__dur)
- constexpr duration **time_since_epoch** () const

Static Public Member Functions

- static constexpr [time_point](#) **max** () noexcept
- static constexpr [time_point](#) **min** () noexcept

Related Symbols

(Note that these are not member symbols.)

- template<typename _Clock , typename _Dur1 , typename _Rep2 , typename _Period2 >
constexpr [time_point](#)< _Clock, typename [common_type](#)< _Dur1, duration< _Rep2, _Period2 > >::type >
[operator+](#) (const [time_point](#)< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)

6.980.1 Detailed Description

```
template<typename _Clock, typename _Dur>
class std::chrono::time_point< _Clock, _Dur >
```

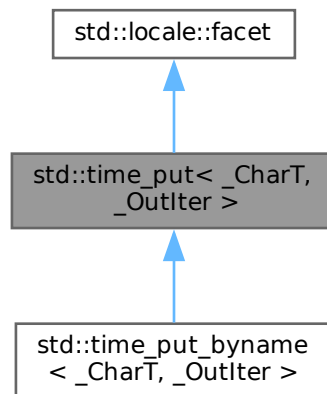
`chrono::time_point` represents a point in time as measured by a clock
 The documentation for this class was generated from the following file:

- [chrono.h](#)

6.981 std::time_put< _CharT, _OutIter > Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for `std::time_put< _CharT, _OutIter >`:



Public Types

- typedef `_CharT` [char_type](#)
- typedef `_OutIter` [iter_type](#)

Public Member Functions

- [time_put](#) (`size_t __refs=0`)
- [iter_type put](#) ([iter_type](#) __s, [ios_base](#) &__io, [char_type](#) __fill, const tm *__tm, char __format, char __mod=0) const
- [iter_type put](#) ([iter_type](#) __s, [ios_base](#) &__io, [char_type](#) __fill, const tm *__tm, const `_CharT` *__beg, const `_CharT` *__end) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual `~time_put()`
- virtual `iter_type do_put(iter_type __s, ios_base &__io, char_type __fill, const tm *__tm, char __format, char __mod) const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale &__cloc) throw()`
- static void `_S_create_c_locale(__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale(__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale()`
- static const char* `_S_get_c_name()` throw()
- static `__c_locale _S_lc_ctype_c_locale(__c_locale __cloc, const char *__s)`

6.981.1 Detailed Description

template<typename _CharT, typename _OutIter>
class std::time_put<_CharT, _OutIter>

Primary class template `time_put`.

This facet encapsulates the code to format and output dates and times according to formats used by `strftime()`.

The `time_put` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `time_put` facet.

6.981.2 Member Typedef Documentation

`char_type`

```
template<typename _CharT, typename _OutIter>
typedef _CharT std::time_put<_CharT, _OutIter>::char_type
```

Public typedefs.

`iter_type`

```
template<typename _CharT, typename _OutIter>
typedef _OutIter std::time_put<_CharT, _OutIter>::iter_type
```

Public typedefs.

6.981.3 Constructor & Destructor Documentation

`time_put()`

```
template<typename _CharT, typename _OutIter>
std::time_put<_CharT, _OutIter>::time_put (
    size_t __refs = 0 ) [inline], [explicit]
```

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

~time_put()

```
template<typename _CharT , typename _OutIter >
virtual std::time_put< _CharT, _OutIter >::~~time_put ( ) [inline], [protected], [virtual]
Destructor.
```

6.981.4 Member Function Documentation**do_put()**

```
template<typename _CharT , typename _OutIter >
_OutIter std::time_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const tm * __tm,
    char __format,
    char __mod ) const [protected], [virtual]
```

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. This function is a hook for derived classes to change the value returned.

See also

put() for more details.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

References [std::ios_base::M_getloc\(\)](#), and [std::__ctype_abstract_base<_CharT>::widen\(\)](#).

Referenced by [std::time_put<_CharT, _OutIter>::put\(\)](#).

put() [1/2]

```
template<typename _CharT , typename _OutIter >
iter_type std::time_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const tm * __tm,
    char __format,
    char __mod = 0 ) const [inline]
```

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. The format and modifier are interpreted as by [strftime\(\)](#). It does so by returning [time_put::do_put\(\)](#).

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__tm</code>	Struct <code>tm</code> with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

References [std::time_put< _CharT, _OutIter >::do_put\(\)](#).

put() [2/2]

```
template<typename _CharT , typename _OutIter >
_OutIter std::time_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const tm * __tm,
    const _CharT * __beg,
    const _CharT * __end ) const
```

Format and output a time or date.

This function formats the data in struct `tm` according to the provided format string. The format string is interpreted as by `strftime()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__tm</code>	Struct <code>tm</code> with date and time info to format.
<code>__beg</code>	Start of format string.
<code>__end</code>	End of format string.

Returns

Iterator after writing.

References [std::ios_base::M_getloc\(\)](#), and [std::__ctype_abstract_base< _CharT >::narrow\(\)](#).

6.981.5 Member Data Documentation**id**

```
template<typename _CharT , typename _OutIter >
locale::id std::time_put< _CharT, _OutIter >::id [static]
```

Numpunct facet id.

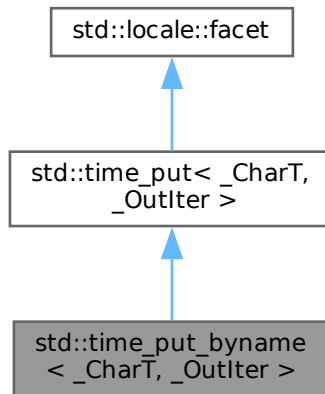
The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

6.982 std::time_put_byname<_CharT, _OutIter> Class Template Reference

```
#include <locale_facets_nonio.h>
```

Inheritance diagram for std::time_put_byname<_CharT, _OutIter>:



Public Types

- typedef _CharT **char_type**
- typedef _OutIter **iter_type**

Public Member Functions

- **time_put_byname** (const char *, size_t __refs=0)
- **time_put_byname** (const [string](#) &__s, size_t __refs=0)
- iter_type **put** (iter_type __s, [ios_base](#) &__io, char_type __fill, const tm *__tm, char __format, char __mod=0) const
- iter_type **put** (iter_type __s, [ios_base](#) &__io, char_type __fill, const tm *__tm, const _CharT *__beg, const _CharT *__end) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual iter_type **do_put** (iter_type __s, [ios_base](#) &__io, char_type __fill, const tm *__tm, char __format, char __mod) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()

- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_type_c_locale** (__c_locale __cloc, const char * __s)

6.982.1 Detailed Description

template<typename _CharT, typename _OutIter>
class std::time_put_byname< _CharT, _OutIter >

class time_put_byname [22.2.5.4].

6.982.2 Member Function Documentation

do_put()

```
template<typename _CharT , typename _OutIter >
_OutIter std::time_put< _CharT, _OutIter >::do_put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const tm * __tm,
    char __format,
    char __mod ) const [protected], [virtual], [inherited]
```

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. This function is a hook for derived classes to change the value returned.

See also

put() for more details.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

References [std::ios_base::M_getloc\(\)](#), and [std::__ctype_abstract_base< _CharT >::widen\(\)](#).

Referenced by [std::time_put< _CharT, _OutIter >::put\(\)](#).

put() [1/2]

```
template<typename _CharT , typename _OutIter >
iter_type std::time_put< _CharT, _OutIter >::put (
    iter_type __s,
    ios_base & __io,
    char_type __fill,
    const tm * __tm,
```

```
char __format,  
char __mod = 0 ) const [inline], [inherited]
```

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. The format and modifier are interpreted as by strftime(). It does so by returning time_put::do_put().

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

References [std::time_put<_CharT, _OutIter>::do_put\(\)](#).

put() [2/2]

```
template<typename _CharT , typename _OutIter >  
_OutIter std::time_put<_CharT, _OutIter>::put (  
    iter_type __s,  
    ios_base & __io,  
    char_type __fill,  
    const tm * __tm,  
    const _CharT * __beg,  
    const _CharT * __end ) const [inherited]
```

Format and output a time or date.

This function formats the data in struct tm according to the provided format string. The format string is interpreted as by strftime().

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__beg</code>	Start of format string.
<code>__end</code>	End of format string.

Returns

Iterator after writing.

References [std::ios_base::_M_getloc\(\)](#), and [std::__ctype_abstract_base<_CharT>::narrow\(\)](#).

6.982.3 Member Data Documentation**id**

```
template<typename _CharT , typename _OutIter >
locale::id std::time\_put< _CharT, _OutIter >::id [static], [inherited]
Numpunct facet id.
```

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

6.983 `std::timed_mutex` Class Reference**Public Member Functions**

- **timed_mutex** (const [timed_mutex](#) &)=delete
- void **lock** ()
- [timed_mutex](#) & **operator=** (const [timed_mutex](#) &)=delete
- bool **try_lock** ()
- template<typename _Rep , typename _Period >
bool **try_lock_for** (const [chrono::duration](#)< _Rep, _Period > &__rtime)
- template<typename _Clock , typename _Duration >
bool **try_lock_until** (const [chrono::time_point](#)< _Clock, _Duration > &__atime)
- void **unlock** ()

6.983.1 Detailed Description

`timed_mutex`

The documentation for this class was generated from the following file:

- [mutex](#)

6.984 `std::to_chars_result` Struct Reference**Public Attributes**

- `errc` **ec**
- `char *` **ptr**

Friends

- bool **operator==** (const [to_chars_result](#) &, const [to_chars_result](#) &)=default

6.984.1 Detailed Description

Result type of `std::to_chars`.

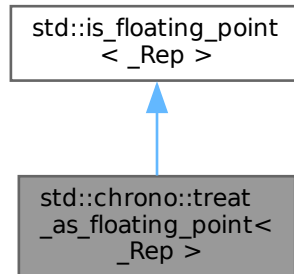
The documentation for this struct was generated from the following file:

- [charconv](#)

6.985 `std::chrono::treat_as_floating_point<_Rep>` Struct Template Reference

```
#include <chrono.h>
```

Inheritance diagram for `std::chrono::treat_as_floating_point<_Rep>`:



6.985.1 Detailed Description

```
template<typename _Rep>
struct std::chrono::treat_as_floating_point<_Rep>
```

Trait indicating whether to treat a type as a floating-point type.

The chrono library uses this trait to tell whether a `duration` can represent fractional values of the given precision, or only integral values.

You should specialize this trait for your own numeric types that are used with `duration` and can represent non-integral values.

Since

C++11

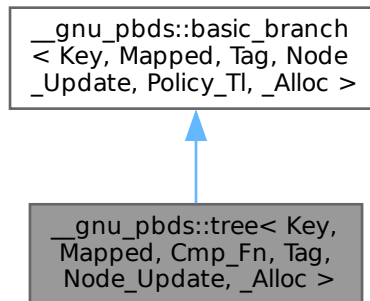
The documentation for this struct was generated from the following file:

- [chrono.h](#)

6.986 `__gnu_pbds::tree<Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc>` Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >`:



Public Types

- typedef Cmp_Fn [cmp_fn](#)
- typedef Node_Update **node_update**

Public Member Functions

- [tree](#) (const [cmp_fn](#) &c)
- **tree** (const [tree](#) &other)
- template<typename It >
[tree](#) (It first, It last)
- template<typename It >
[tree](#) (It first, It last, const [cmp_fn](#) &c)
- [tree](#) & **operator=** (const [tree](#) &other)
- void **swap** ([tree](#) &other)

6.986.1 Detailed Description

```

template<typename Key, typename Mapped, typename Cmp_Fn = std::less<Key>, typename Tag = rb_↵
tree_tag, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class
Node_Update = null_node_update, typename _Alloc = std::allocator<char>>
class __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >
  
```

A tree-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Cmp_Fn</i>	Comparison functor.
<i>Tag</i>	Instantiating data structure type, see <code>container_tag</code> .
<i>Node_Update</i>	Updates tree internal-nodes, restores invariants when invalidated. XXX See <code>design::tree-based-containersnode</code> invariants.
<i>_Alloc</i>	Allocator type.

Base tag choices are: `ov_tree_tag`, `rb_tree_tag`, `splay_tree_tag`.
Base is `basic_branch`.

6.986.2 Member Typedef Documentation

`cmp_fn`

```
template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb<
_tree_tag, template< typename Node_CItr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ >
class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>>
typedef Cmp_Fn \_\_gnu\_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::cmp_fn
Comparison functor type.
```

6.986.3 Constructor & Destructor Documentation

`tree()` [1/3]

```
template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb<
_tree_tag, template< typename Node_CItr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ >
class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>>
\_\_gnu\_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::tree (
    const cmp_fn & c ) [inline]
```

Constructor taking some policy objects. `r_cmp_fn` will be copied by the `Cmp_Fn` object of the container object.

`tree()` [2/3]

```
template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb<
_tree_tag, template< typename Node_CItr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ >
class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>>
template<typename It >
\_\_gnu\_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::tree (
    It first,
    It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

`tree()` [3/3]

```
template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb<
_tree_tag, template< typename Node_CItr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ >
class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>>
template<typename It >
\_\_gnu\_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::tree (
    It first,
    It last,
    const cmp_fn & c ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_cmp_fn` will be copied by the `cmp_fn` object of the container object.
The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.987 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >` Struct Template Reference

6.987.1 Detailed Description

```
template<typename Node_Update, bool _BTp>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >
```

Tree metadata helper.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

6.988 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >` Struct Template Reference

```
#include <node_metadata_selector.hpp>
```

Public Types

- typedef `Node_Update::metadata_type` `type`

6.988.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, false >
```

Specialization, false.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

6.989 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >` Struct Template Reference

```
#include <node_metadata_selector.hpp>
```

Public Types

- typedef `null_type` `type`

6.989.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, true >
```

Specialization, true.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

6.990 `__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >` Struct Template Reference

```
#include <node_metadata_selector.hpp>
```


Public Types

- typedef [tree_metadata_helper](#)< __node_u, null_update >::type **type**

6.990.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename
Const_Itrator, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >
```

Tree node metadata dispatch.

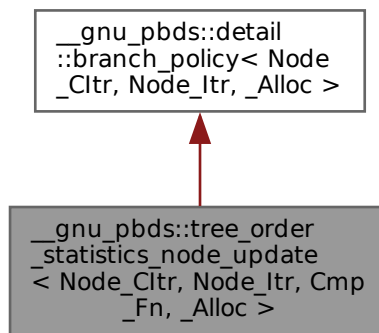
The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

6.991 __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > Class Template Reference

```
#include <tree_policy.hpp>
```

Inheritance diagram for __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >:



Public Types

- typedef _Alloc **allocator_type**
- typedef Cmp_Fn **cmp_fn**
- typedef node_const_iterator::value_type **const_iterator**
- typedef node_iterator::value_type **iterator**
- typedef base_type::key_const_reference **key_const_reference**
- typedef base_type::key_type **key_type**
- typedef size_type **metadata_type**
- typedef Node_Cltr **node_const_iterator**
- typedef Node_Itr **node_iterator**
- typedef allocator_type::size_type **size_type**

Public Member Functions

- iterator `find_by_order` (size_type)
- const_iterator `find_by_order` (size_type) const
- size_type `order_of_key` (key_const_reference) const

Protected Member Functions

- void `operator()` (node_iterator, node_const_iterator) const

6.991.1 Detailed Description

`template<typename Node_CIttr, typename Node_Itr, typename Cmp_Fn, typename _Alloc>`
`class __gnu_pbds::tree_order_statistics_node_update< Node_CIttr, Node_Itr, Cmp_Fn, _Alloc >`

Functor updating ranks of entrees.

6.991.2 Member Function Documentation

`find_by_order()` [1/2]

```
template<typename Node_CIttr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
iterator __gnu_pbds::tree_order_statistics_node_update< Node_CIttr, Node_Itr, Cmp_Fn, _Alloc >↵
::find_by_order (
    size_type ) [inline]
```

Finds an entry by `__order`. Returns an iterator to the entry with the `__order` order, or an iterator to the container object's end if order is at least the size of the container object.

`find_by_order()` [2/2]

```
template<typename Node_CIttr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
const_iterator __gnu_pbds::tree_order_statistics_node_update< Node_CIttr, Node_Itr, Cmp_Fn, _Alloc
>::find_by_order (
    size_type ) const [inline]
```

Finds an entry by `__order`. Returns a const_iterator to the entry with the `__order` order, or a const_iterator to the container object's end if order is at least the size of the container object.

`operator()`

```
template<typename Node_CIttr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
void __gnu_pbds::tree_order_statistics_node_update< Node_CIttr, Node_Itr, Cmp_Fn, _Alloc >::operator()
(
    node_iterator ,
    node_const_iterator ) const [inline], [protected]
```

Updates the rank of a node through a node_iterator `node_it`; `end_nd_it` is the end node iterator.

`order_of_key()`

```
template<typename Node_CIttr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
size_type __gnu_pbds::tree_order_statistics_node_update< Node_CIttr, Node_Itr, Cmp_Fn, _Alloc >↵
::order_of_key (
    key_const_reference ) const [inline]
```

Returns the order of a key within a sequence. For exapmle, if `r_key` is the smallest key, this method will return 0; if `r_key` is a key between the smallest and next key, this method will return 1; if `r_key` is a key larger than the largest key, this method will return the size of `r_c`.

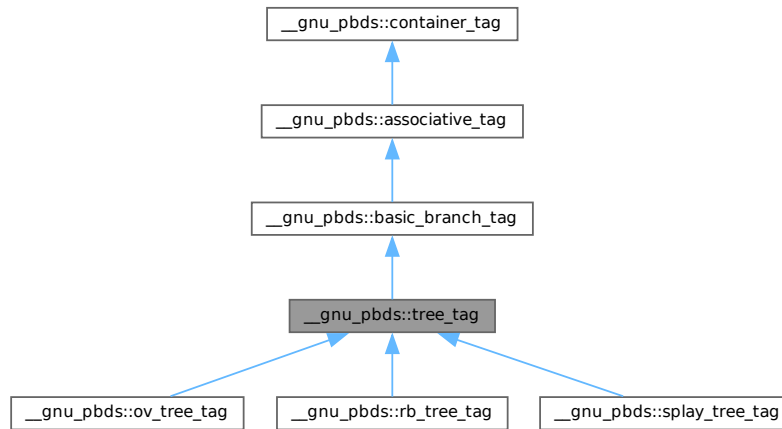
The documentation for this class was generated from the following file:

- [tree_policy.hpp](#)

6.992 __gnu_pbds::tree_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for __gnu_pbds::tree_tag:



6.992.1 Detailed Description

Basic tree structure.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.993 __gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc > Struct Template Reference

6.993.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename
Node_Itr, typename Cmp_Fn_, typename _Alloc > class Node_Update, typename Tag, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc >
```

Tree traits class, primary template.

The documentation for this struct was generated from the following file:

- [branch_policy/traits.hpp](#)

6.994 __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc > Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef [tree_node_metadata_dispatch](#)< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type **metadata_type**

- typedef `ov_tree_node_const_it_< value_type, metadata_type, _Alloc >` `node_const_iterator`
- typedef `ov_tree_node_it_< value_type, metadata_type, _Alloc >` `node_iterator`
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` `node_update`
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`

6.994.1 Detailed Description

```
template<typename Key, typename Mapped, class Cmp_Fn, template< typename Node_CItr, class Node_Itr,
class Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >
```

Tree traits.

6.994.2 Member Typedef Documentation

`node_const_iterator`

```
template<typename Key , typename Mapped , class Cmp_Fn , template< typename Node_CItr, class
Node_Itr, class Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef ov_tree_node_const_it_< value_type, metadata_type, _Alloc> __gnu_pbds::detail::tree_traits<
Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

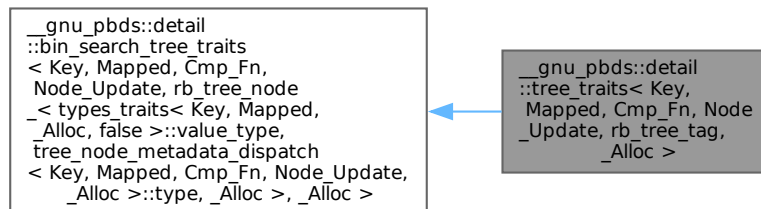
The documentation for this struct was generated from the following file:

- [ov_tree_map_/traits.hpp](#)

6.995 `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >` Struct Template Reference

```
#include <traits.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`:



Public Types

- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` `const_reverse_iterator`
- typedef `rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` `node`

- typedef `bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc > node_const_iterator`
- typedef `bin_search_tree_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc > node_iterator`
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > node_update`
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`
- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > point_const_iterator`
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > point_iterator`
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc > reverse_iterator`

6.995.1 Detailed Description

`template<typename Key, typename Mapped, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc > class Node_Update, typename _Alloc>`

`struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`

Specialization.

6.995.2 Member Typedef Documentation

`node_const_iterator`

```
typedef bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false
>::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits<
Key, Mapped, Cmp_Fn, Node_Update, rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, _Alloc >::node_const_iterator [inherited]
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

The documentation for this struct was generated from the following file:

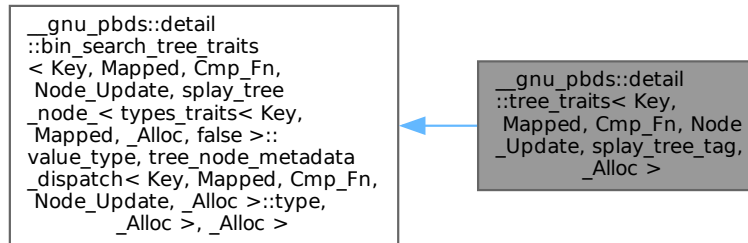
- [rb_tree_map/traits.hpp](#)

6.996 `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >` Struct Template Reference

```
#include <traits.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc`

>:



Public Types

- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **const_reverse_iterator**
- typedef `splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **node**
- typedef `bin_search_tree_const_node_it_< splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc >` **node_const_iterator**
- typedef `bin_search_tree_node_it_< splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`
- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_const_iterator**
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_iterator**
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **reverse_iterator**

6.996.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
```

```
struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >
```

Specialization.

6.996.2 Member Typedef Documentation

node_const_iterator

```
typedef bin_search_tree_const_node_it_< splay_tree_node_< types_traits< Key, Mapped, _Alloc,
false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >↵
::type, _Alloc > , point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits<
Key, Mapped, Cmp_Fn, Node_Update, splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >↵
::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _↵
Alloc > , _Alloc >::node_const_iterator [inherited]
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

The documentation for this struct was generated from the following file:

- [splay_tree_/traits.hpp](#)

6.997 __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc > Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef [tree_node_metadata_dispatch](#)< Key, [null_type](#), Cmp_Fn, Node_Update, _Alloc >::type [metadata_type](#)
- typedef [ov_tree_node_const_it_](#)< [value_type](#), [metadata_type](#), _Alloc > [node_const_iterator](#)
- typedef [node_const_iterator](#) [node_iterator](#)
- typedef Node_Update< [node_const_iterator](#), [node_const_iterator](#), Cmp_Fn, _Alloc > [node_update](#)
- typedef [__gnu_pbds::null_node_update](#)< [node_const_iterator](#), [node_iterator](#), Cmp_Fn, _Alloc > * [null_node_↵](#)
[update_pointer](#)

6.997.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_CItr, class Node_Itr, class Cmp_Fn_↵
, typename _Alloc_ > class Node_Update, typename _Alloc>
```

```
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >
```

Specialization.

6.997.2 Member Typedef Documentation

node_const_iterator

```
template<typename Key , class Cmp_Fn , template< typename Node_CItr, class Node_Itr, class Cmp_↵
Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef ov_tree_node_const_it_< value\_type, metadata\_type, _Alloc> __gnu_pbds::detail::tree_traits<
Key, null\_type, Cmp_Fn, Node_Update, ov\_tree\_tag, _Alloc >::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

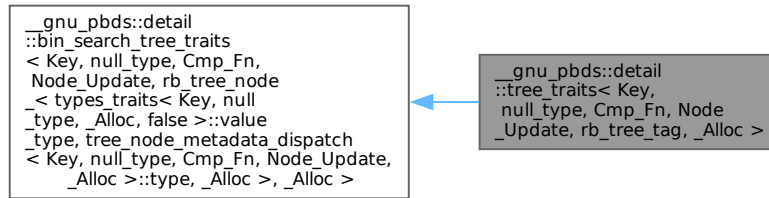
The documentation for this struct was generated from the following file:

- [ov_tree_map_/traits.hpp](#)

6.998 __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc > Struct Template Reference

```
#include <traits.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`:



Public Types

- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **const_reverse_iterator**
- typedef `rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **node**
- typedef `bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc >` **node_const_iterator**
- typedef `bin_search_tree_node_it_< rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`
- typedef `bin_search_tree_const_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_const_iterator**
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_iterator**
- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **reverse_iterator**

6.998.1 Detailed Description

```

template<typename Key, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >
    
```

Specialization.

6.998.2 Member Typedef Documentation

node_const_iterator

```

typedef bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::
    
```



```

::type, _Alloc > , point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits<
Key, null_type , Cmp_Fn, Node_Update, rb_tree_node< types_traits< Key, null_type, _Alloc, false
>::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type,
_Alloc > , _Alloc >::node_const_iterator [inherited]

```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

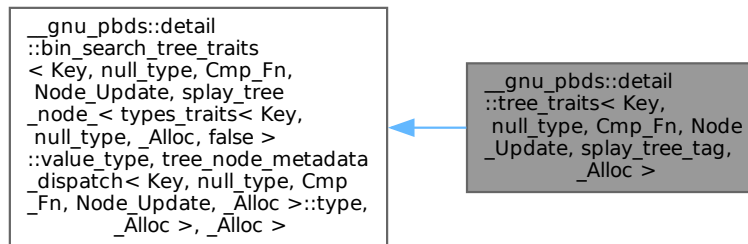
The documentation for this struct was generated from the following file:

- [rb_tree_map_traits.hpp](#)

6.999 __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc > Struct Template Reference

```
#include <traits.hpp>
```

Inheritance diagram for __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >:



Public Types

- typedef `bin_search_tree_const_it` < typename `node_alloc_traits::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, false, `_Alloc` > **const_reverse_iterator**
- typedef `splay_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` > **node**
- typedef `bin_search_tree_const_node_it` < `splay_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_const_iterator**
- typedef `bin_search_tree_node_it` < `splay_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_iterator**
- typedef `Node_Update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > **node_update**
- typedef `__gnu_pbds::null_node_update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > * **null_node_update_pointer**
- typedef `bin_search_tree_const_it` < typename `node_alloc_traits::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_const_iterator**
- typedef `bin_search_tree_it` < typename `node_alloc_traits::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_iterator**

- typedef `bin_search_tree_it_< typename node_alloc_traits::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **reverse_iterator**

6.999.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_↵
, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >
```

Specialization.

6.999.2 Member Typedef Documentation

`node_const_iterator`

```
typedef bin_search_tree_const_node_it_< splay_tree_node_< types_traits< Key, null_type, _Alloc,
false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >↵
::type, _Alloc > , point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits<
Key, null_type , Cmp_Fn, Node_Update, splay_tree_node_< types_traits< Key, null_type, _Alloc,
false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >↵
::type, _Alloc > , _Alloc >::node_const_iterator [inherited]
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

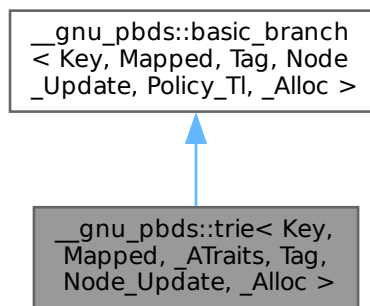
The documentation for this struct was generated from the following file:

- [splay_tree_/traits.hpp](#)

6.1000 `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >` Class Template Reference

```
#include <assoc_container.hpp>
```

Inheritance diagram for `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >`:



Public Types

- typedef `_ATraits` `access_traits`
- typedef `Node_Update` `node_update`

Public Member Functions

- `trie` (const `access_traits` &t)
- `trie` (const `trie` &other)
- `template<typename It >`
`trie` (It first, It last)
- `template<typename It >`
`trie` (It first, It last, const `access_traits` &t)
- `trie & operator=` (const `trie` &other)
- `void swap` (`trie` &other)

6.1000.1 Detailed Description

```
template<typename Key, typename Mapped, typename _ATraits = typename detail::default_trie_access_↵
traits<Key>::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename Node_Itr, type-
name _ATraits_, typename _Alloc_ > class Node_Update = null_node_update, typename _Alloc = std.↵
::allocator<char>>>
```

```
class __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >
```

A trie-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>_ATraits</i>	Element access traits.
<i>Tag</i>	Instantiating data structure type, see <code>container_tag</code> .
<i>Node_Update</i>	Updates trie internal-nodes, restores invariants when invalidated. XXX See <code>design::tree-based-containersnode</code> invariants.
<i>_Alloc</i>	Allocator type.

Base tag choice is `pat_trie_tag`.

Base is `basic_branch`.

6.1000.2 Member Typedef Documentation

`access_traits`

```
template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_↵
access_traits<Key>::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename
Node_Itr, typename _ATraits_, typename _Alloc_ > class Node_Update = null_node_update, typename
_Alloc = std::allocator<char>>>
```

```
typedef _ATraits __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::access_↵
traits
```

Element access traits type.

6.1000.3 Constructor & Destructor Documentation

`trie()` [1/3]

```
template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_↵
access_traits<Key>::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename
Node_Itr, typename _ATraits_, typename _Alloc_ > class Node_Update = null_node_update, typename
_Alloc = std::allocator<char>>>
```

```
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie (
    const access_traits & t ) [inline]
```

Constructor taking some policy objects. `r_access_traits` will be copied by the `_ATraits` object of the container object.

`trie()` [2/3]

```
template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_↵
access_traits<Key>::type, typename Tag = pat_trie_tag, template< typename Node_CIttr, typename
Node_Itr, typename _ATraits_, typename _Alloc_ > class Node_Update = null_node_update, typename
_Alloc = std::allocator<char>>>
template<typename It >
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie (
    It first,
    It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

`trie()` [3/3]

```
template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_↵
access_traits<Key>::type, typename Tag = pat_trie_tag, template< typename Node_CIttr, typename
Node_Itr, typename _ATraits_, typename _Alloc_ > class Node_Update = null_node_update, typename
_Alloc = std::allocator<char>>>
template<typename It >
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie (
    It first,
    It last,
    const access_traits & t ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

6.1001 `__gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >` Struct Template Reference

6.1001.1 Detailed Description

```
template<typename Node_Update, bool _BTp>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >
```

Trie metadata helper.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

6.1002 `__gnu_pbds::detail::trie_metadata_helper< Node_Update, false >` Struct Template Reference

```
#include <node_metadata_selector.hpp>
```

Public Types

- `typedef Node_Update::metadata_type type`

6.1002.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, false >
```

Specialization, false.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

6.1003 __gnu_pbds::detail::trie_metadata_helper< Node_Update, true > Struct Template Reference

```
#include <node_metadata_selector.hpp>
```

Public Types

- typedef [null_type](#) type

6.1003.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, true >
```

Specialization, true.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

6.1004 __gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc > Struct Template Reference

```
#include <node_metadata_selector.hpp>
```

Public Types

- typedef [trie_metadata_helper](#)< __node_u, null_update >::type type

6.1004.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename
Const_Iterator, typename Cmp_Fn_, typename _Alloc > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >
```

Trie node metadata dispatch.

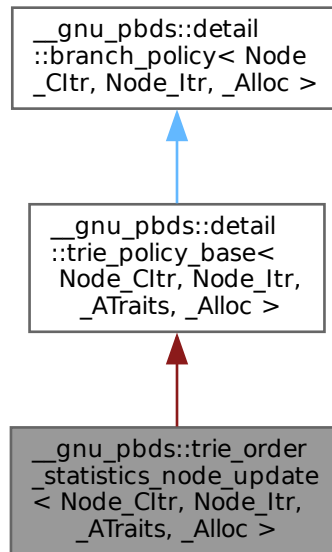
The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

6.1005 __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference

```
#include <trie_policy.hpp>
```

Inheritance diagram for __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >:



Public Types

- typedef `access_traits::const_iterator` **a_const_iterator**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef `size_type` **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` **size_type**

Public Member Functions

- iterator `find_by_order` (`size_type`)
- const_iterator `find_by_order` (`size_type`) const
- size_type `order_of_key` (`key_const_reference`) const
- size_type `order_of_prefix` (`a_const_iterator`, `a_const_iterator`) const

Protected Member Functions

- void `operator()` (`node_iterator`, `node_const_iterator`) const

Private Member Functions

- virtual const_iterator **end** () const =0
- virtual const access_traits & **get_access_traits** () const =0

6.1005.1 Detailed Description

```
template<typename Node_CIttr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _Alloc >
```

Functor updating ranks of entrees.

6.1005.2 Member Function Documentation

find_by_order() [1/2]

```
template<typename Node_CIttr , typename Node_Itr , typename _ATraits , typename _Alloc >
iterator __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _Alloc >↵
::find_by_order (
    size_type ) [inline]
```

Finds an entry by `__order`. Returns an iterator to the entry with the `__order` order, or an iterator to the container object's end if order is at least the size of the container object.

find_by_order() [2/2]

```
template<typename Node_CIttr , typename Node_Itr , typename _ATraits , typename _Alloc >
const_iterator __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _↵
_Alloc >::find_by_order (
    size_type ) const [inline]
```

Finds an entry by `__order`. Returns a `const_iterator` to the entry with the `__order` order, or a `const_iterator` to the container object's end if order is at least the size of the container object.

operator>()()

```
template<typename Node_CIttr , typename Node_Itr , typename _ATraits , typename _Alloc >
void __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _Alloc >↵
::operator() (
    node_iterator ,
    node_const_iterator ) const [inline], [protected]
```

Updates the rank of a node through a `node_iterator` `node_it`; `end_nd_it` is the end node iterator.

order_of_key()

```
template<typename Node_CIttr , typename Node_Itr , typename _ATraits , typename _Alloc >
size_type __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _Alloc >↵
::order_of_key (
    key_const_reference ) const [inline]
```

Returns the order of a key within a sequence. For exapmle, if `r_key` is the smallest key, this method will return 0; if `r_key` is a key between the smallest and next key, this method will return 1; if `r_key` is a key larger than the largest key, this method will return the size of `r_c`.

order_of_prefix()

```
template<typename Node_CIttr , typename Node_Itr , typename _ATraits , typename _Alloc >
size_type __gnu_pbds::trie_order_statistics_node_update< Node_CIttr, Node_Itr, _ATraits, _Alloc >↵
::order_of_prefix (
```

```

    a_const_iterator ,
    a_const_iterator ) const [inline]

```

Returns the order of a prefix within a sequence. For example, if [b, e] is the smallest prefix, this method will return 0; if r_key is a key between the smallest and next key, this method will return 1; if r_key is a key larger than the largest key, this method will return the size of r_c.

The documentation for this class was generated from the following file:

- [trie_policy.hpp](#)

6.1006 `__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >` Class Template Reference

```
#include <trie_policy_base.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >`:



Public Types

- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef `null_type` **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` **size_type**

Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Itr::value_type` **it_type**
- typedef `remove_const< key_type >::type` **rckey_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `rebind_traits< _Alloc, rckey_type >` **rebind_k**
- typedef `rebind_traits< _Alloc, rcvalue_type >` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual const_iterator **end** () const =0
- virtual iterator **end** ()=0
- it_type **end_iterator** () const
- virtual const_access_traits & **get_access_traits** () const =0
- virtual node_const_iterator **node_begin** () const =0
- virtual node_iterator **node_begin** ()=0
- virtual node_const_iterator **node_end** () const =0
- virtual node_iterator **node_end** ()=0

Static Protected Member Functions

- static size_type **common_prefix_len** (node_iterator, e_const_iterator, e_const_iterator, const access_traits &)
- static key_const_reference **extract_key** (const_reference r_val)
- static iterator **leftmost_it** (node_iterator)
- static bool **less** (e_const_iterator, e_const_iterator, e_const_iterator, e_const_iterator, const access_traits &)
- static iterator **rightmost_it** (node_iterator)

6.1006.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

Base class for trie policies.

6.1006.2 Member Function Documentation

end()

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
virtual iterator __gnu_pbds::detail::trie_policy_base< Node_CItr, Node_Itr, _ATraits, _Alloc >↔
::end ( ) [protected], [pure virtual]
```

Implements [__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >](#).

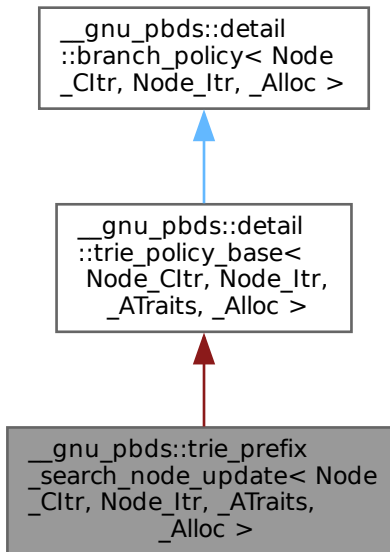
The documentation for this class was generated from the following file:

- [trie_policy_base.hpp](#)

6.1007 __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference

```
#include <trie_policy.hpp>
```

Inheritance diagram for `__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >`:



Public Types

- typedef `access_traits::const_iterator` [a_const_iterator](#)
- typedef `_ATraits` [access_traits](#)
- typedef `_Alloc` [allocator_type](#)
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef [null_type](#) **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` [size_type](#)

Public Member Functions

- `std::pair< iterator, iterator >` [prefix_range](#) ([a_const_iterator](#), [a_const_iterator](#))
- `std::pair< const_iterator, const_iterator >` [prefix_range](#) ([a_const_iterator](#), [a_const_iterator](#)) const
- `std::pair< iterator, iterator >` [prefix_range](#) (`key_const_reference`)
- `std::pair< const_iterator, const_iterator >` [prefix_range](#) (`key_const_reference`) const

Protected Member Functions

- void [operator\(\)](#) (`node_iterator` `node_it`, `node_const_iterator` `end_nd_it`) const

6.1007.1 Detailed Description

```
template<typename Node_CItr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc >
```

A node updator that allows tries to be searched for the range of values that match a certain prefix.

6.1007.2 Member Typedef Documentation

a_const_iterator

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
typedef access_traits::const_iterator __gnu_pbds::trie_prefix_search_node_update< Node_CItr,
Node_Itr, _ATraits, _Alloc >::a_const_iterator
Const element iterator.
```

access_traits

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
typedef _ATraits __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _↵
Alloc >::access_traits
Element access traits.
```

allocator_type

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
typedef _Alloc __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc
>::allocator_type
_Alloc type.
```

size_type

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
typedef allocator_type::size_type __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_↵
Itr, _ATraits, _Alloc >::size_type
Size type.
```

6.1007.3 Member Function Documentation

operator>()

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
void __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc >::operator()
(
    node_iterator node_it,
    node_const_iterator end_nd_it ) const [inline], [protected]
```

Called to update a node's metadata.

prefix_range() [1/4]

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
std::pair< iterator, iterator > __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr,
_ATraits, _Alloc >::prefix_range (
    a_const_iterator ,
    a_const_iterator )
```

Finds the iterator range corresponding to all values whose prefixes match [b, e).

prefix_range() [2/4]

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
std::pair< const_iterator, const_iterator > __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc >::prefix_range (
    a_const_iterator ,
    a_const_iterator ) const
```

Finds the const iterator range corresponding to all values whose prefixes match [b, e).

prefix_range() [3/4]

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
std::pair< iterator, iterator > __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc >::prefix_range (
    key_const_reference )
```

Finds the iterator range corresponding to all values whose prefixes match r_key.

prefix_range() [4/4]

```
template<typename Node_CItr , typename Node_Itr , typename _ATraits , typename _Alloc >
std::pair< const_iterator, const_iterator > __gnu_pbds::trie_prefix_search_node_update< Node_CItr, Node_Itr, _ATraits, _Alloc >::prefix_range (
    key_const_reference ) const
```

Finds the const iterator range corresponding to all values whose prefixes match r_key.

The documentation for this class was generated from the following file:

- [trie_policy.hpp](#)

6.1008 `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc > Struct` Template Reference

```
#include <trie_policy.hpp>
```

Public Types

- enum { **reverse** }
- enum { **min_e_val** , **max_e_val** , **max_size** }
- typedef detail::__conditional_type< Reverse, typenameString::const_reverse_iterator, typenameString::const_iterator >::__type **const_iterator**
- typedef [std::iterator_traits< const_iterator >::value_type](#) **e_type**
- typedef [detail::rebind_traits< _Alloc, key_type >::const_reference](#) **key_const_reference**
- typedef String **key_type**
- typedef _Alloc::size_type **size_type**

Static Public Member Functions

- static [const_iterator](#) **begin** (key_const_reference)
- static size_type **e_pos** ([e_type](#) e)
- static [const_iterator](#) **end** (key_const_reference)

6.1008.1 Detailed Description

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__numeric_↵
traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail::__numeric_↵
traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc = std::allocator<char>>
struct __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >
```

Element access traits for string types.

Template Parameters

<i>String</i>	String type.
<i>Min_E_Val</i>	Minimal element value.
<i>Max_E_Val</i>	Maximum element value.
<i>Reverse</i>	Reverse iteration should be used. Default: false.
<i>_Alloc</i>	Allocator type.

6.1008.2 Member Typedef Documentation

const_iterator

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__↵
numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail↵
::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc =
std::allocator<char>>
typedef detail::__conditional_type<Reverse,typenameString::const_reverse_iterator,typenameString↵
::const_iterator>::__type __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val,
Reverse, _Alloc >::__const_iterator
```

Element const iterator type.

e_type

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__↵
numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail↵
::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc =
std::allocator<char>>
typedef std::iterator_traits<const_iterator>::value_type __gnu_pbds::trie_string_access_traits<
String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::e_type
```

Element type.

6.1008.3 Member Function Documentation

begin()

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__↵
numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail↵
::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc =
std::allocator<char>>
static const_iterator __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse,
_Alloc >::begin (
    key_const_reference ) [inline], [static]
```

Returns a const_iterator to the first element of key_const_reference agumnet.

e_pos()

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__↵
numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail↵
::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc =
std::allocator<char>>
static size_type __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _↵
Alloc >::e_pos (
    e_type e ) [inline], [static]
```

Maps an element to a position.

end()

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__↵
numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail↵
::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc =
std::allocator<char>>
static const_iterator __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse,
_Alloc >::end (
    key_const_reference ) [inline], [static]
```

Returns a const_iterator to the after-last element of key_const_reference argument.

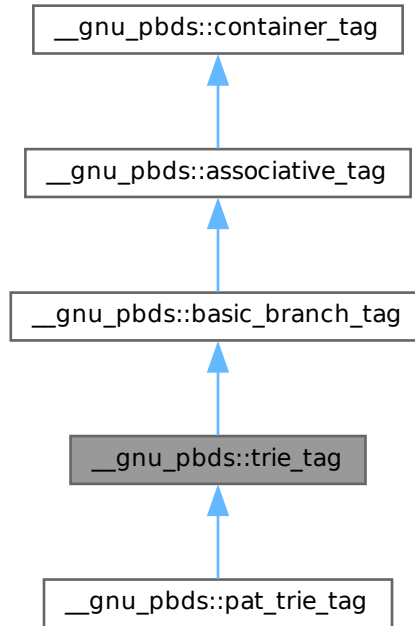
The documentation for this struct was generated from the following file:

- [trie_policy.hpp](#)

6.1009 __gnu_pbds::trie_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

Inheritance diagram for `__gnu_pbds::trie_tag`:



6.1009.1 Detailed Description

Basic trie structure.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.1010 `__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >` Struct Template Reference

6.1010.1 Detailed Description

```
template<typename Key, typename Data, typename _ATraits, template< typename Node_Cltr, typename
Node_Itr, typename _ATraits_, typename _Alloc > class Node_Update, typename Tag, typename _Alloc>
struct __gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >
```

Trie traits class, primary template.

The documentation for this struct was generated from the following file:

- [branch_policy/traits.hpp](#)

6.1011 `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >` Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef `_ATraits` **access_traits**
- typedef `base_type::Clter< node, leaf, head, inode, true >` **const_iterator**
- typedef `base_type::Clter< node, leaf, head, inode, false >` **const_reverse_iterator**
- typedef `base_type::Head< synth_access_traits, metadata >` **head**
- typedef `base_type::Inode< synth_access_traits, metadata >` **inode**
- typedef `base_type::Iter< node, leaf, head, inode, true >` **iterator**
- typedef `base_type::Leaf< synth_access_traits, metadata >` **leaf**
- typedef `base_type::Metadata< metadata_type, _Alloc >` **metadata**
- typedef `trie_node_metadata_dispatch< Key, Mapped, _ATraits, Node_Update, _Alloc >::type` **metadata_type**
- typedef `base_type::Node_base< synth_access_traits, metadata >` **node**
- typedef `base_type::Node_citer< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_const_iterator**
- typedef `base_type::Node_iter< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, _ATraits, _Alloc >` **node_update**
- typedef `null_node_update< node_const_iterator, node_iterator, _ATraits, _Alloc > *` **null_node_update_pointer**
- typedef `base_type::Iter< node, leaf, head, inode, false >` **reverse_iterator**
- typedef `__gnu_pbds::detail::synth_access_traits< type_traits, false, access_traits >` **synth_access_traits**

6.1011.1 Detailed Description

```
template<typename Key, typename Mapped, typename _ATraits, template< typename Node_CItr, typename
Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >
```

Specialization.

6.1011.2 Member Typedef Documentation

node_const_iterator

```
template<typename Key , typename Mapped , typename _ATraits , template< typename Node_CItr, typename
Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef base_type::Node_citer<node, leaf, head, inode, const_iterator, iterator, _Alloc> __gnu_pbds::detail::tr
Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >::node_const_iterator
This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.
```

node_update

```
template<typename Key , typename Mapped , typename _ATraits , template< typename Node_CItr, typename
Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef Node_Update<node_const_iterator, node_iterator, _ATraits, _Alloc> __gnu_pbds::detail::trie_traits<
Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >::node_update
Type for node update.
```

synth_access_traits

```
template<typename Key , typename Mapped , typename _ATraits , template< typename Node_CItr, typename
Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef __gnu_pbds::detail::synth_access_traits<type_traits, false, access_traits> __gnu_pbds::detail::trie_traits<
Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >::synth_access_traits
Type for synthesized traits.
```

The documentation for this struct was generated from the following file:

- [pat_trie_/traits.hpp](#)

6.1012 `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >` Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- typedef `_ATraits` **access_traits**
- typedef `base_type::Cltr< node, leaf, head, inode, true >` **const_iterator**
- typedef `base_type::Cltr< node, leaf, head, inode, false >` **const_reverse_iterator**
- typedef `base_type::Head< synth_access_traits, metadata >` **head**
- typedef `base_type::Inode< synth_access_traits, metadata >` **inode**
- typedef `const_iterator` **iterator**
- typedef `base_type::Leaf< synth_access_traits, metadata >` **leaf**
- typedef `base_type::Metadata< metadata_type, _Alloc >` **metadata**
- typedef `trie_node_metadata_dispatch< Key, null_type, _ATraits, Node_Update, _Alloc >::type` **metadata_type**
- typedef `base_type::Node_base< synth_access_traits, metadata >` **node**
- typedef `base_type::Node_citer< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_const_iterator**
- typedef `node_const_iterator` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, _ATraits, _Alloc >` **node_update**
- typedef `null_node_update< node_const_iterator, node_const_iterator, _ATraits, _Alloc > * null_node_update` **←_pointer**
- typedef `const_reverse_iterator` **reverse_iterator**
- typedef `__gnu_pbds::detail::synth_access_traits< type_traits, true, access_traits >` **synth_access_traits**

6.1012.1 Detailed Description

```
template<typename Key, typename _ATraits, template< typename Node_CItr, typename Node_Itr, typename
Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >
```

Specialization.

6.1012.2 Member Typedef Documentation

node_const_iterator

```
template<typename Key , typename _ATraits , template< typename Node_CItr, typename Node_Itr,
typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef base_type::Node_citer<node, leaf, head, inode, const_iterator, iterator, _Alloc> __gnu_pbds::detail::tr
Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

node_update

```
template<typename Key , typename _ATraits , template< typename Node_CItr, typename Node_Itr,
typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef Node_Update<node_const_iterator, node_iterator, _ATraits, _Alloc> __gnu_pbds::detail::trie_traits<
Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >::node_update
```

Type for node update.

synth_access_traits

```
template<typename Key , typename _ATraits , template< typename Node_CItr, typename Node_Itr,
typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc >
typedef __gnu_pbds::detail::synth_access_traits<type_traits, true, access_traits> __gnu_pbds::detail::trie_traits<
Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >::synth_access_traits
```

Type for synthesized traits.

The documentation for this struct was generated from the following file:

- [pat_trie_/traits.hpp](#)

6.1013 __gnu_pbds::trivial_iterator_tag Struct Reference

```
#include <tag_and_trait.hpp>
```

6.1013.1 Detailed Description

A trivial iterator tag. Signifies that the iterators has none of std::iterators's movement abilities.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

6.1014 std::try_to_lock_t Struct Reference

```
#include <std_mutex.h>
```

6.1014.1 Detailed Description

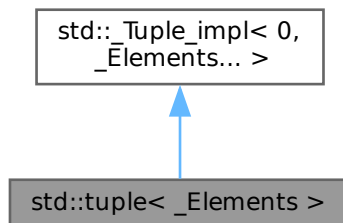
Try to acquire ownership of the mutex without blocking.

The documentation for this struct was generated from the following file:

- [std_mutex.h](#)

6.1015 std::tuple<_Elements> Class Template Reference

Inheritance diagram for std::tuple<_Elements>:

**Public Member Functions**

- `template<typename... _UElements, bool _Valid = __valid_args<_UElements...>(), _ImplicitCtor<_Valid, _UElements...> = true>`
`constexpr tuple (_UElements &&... __elements) noexcept(__nothrow_constructible<_UElements...>())`

- `template<typename... _UElements, bool _Valid = __valid_args<_UElements...>(), _ExplicitCtor<_Valid, _UElements...> = false>
constexpr tuple (_UElements &&... __elements) noexcept(__nothrow_constructible<_UElements...>())`
- `template<typename _Alloc, _ImplicitDefaultCtor<is_object<_Alloc>::value> = true>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a)
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, _UElements &&... __elements)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = __valid_args<_UElements...>(), _ImplicitCtor<_Valid, _UElements...> = true>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, _UElements &&... __elements)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = __valid_args<_UElements...>(), _ExplicitCtor<_Valid, _UElements...> = false>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, _UElements &&... __elements)`
- `template<typename _Alloc, bool _NotEmpty = (sizeof...(Elements) >= 1), _ImplicitCtor<_NotEmpty, const Elements &...> = true>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, const Elements &... __elements)`
- `template<typename _Alloc, bool _NotEmpty = (sizeof...(Elements) >= 1), _ExplicitCtor<_NotEmpty, const Elements &...> = false>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, const Elements &... __elements)`
- `template<typename _Alloc >
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, const tuple &__in)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_<←
ctor<const tuple<_UElements...>&&>(), _ImplicitCtor<_Valid, const _UElements &...> = true>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, const tuple<_UElements...> &__in)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_<←
ctor<const tuple<_UElements...>&&>(), _ExplicitCtor<_Valid, const _UElements &...> = false>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, const tuple<_UElements...> &__in)`
- `template<typename _Alloc >
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, tuple &&__in)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_<←
ctor<tuple<_UElements...>&&>(), _ImplicitCtor<_Valid, _UElements...> = true>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, tuple<_UElements...> &&__in)`
- `template<typename _Alloc, typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_<←
ctor<tuple<_UElements...>&&>(), _ExplicitCtor<_Valid, _UElements...> = false>
constexpr tuple (allocator_arg_t __tag, const _Alloc &__a, tuple<_UElements...> &&__in)`
- `template<bool _NotEmpty = (sizeof...(Elements) >= 1), _ImplicitCtor<_NotEmpty, const Elements &...> = true>
constexpr tuple (const Elements &... __elements) noexcept(__nothrow_constructible<const Elements &...>())`
- `template<bool _NotEmpty = (sizeof...(Elements) >= 1), _ExplicitCtor<_NotEmpty, const Elements &...> = false>
constexpr tuple (const Elements &... __elements) noexcept(__nothrow_constructible<const Elements &...>())`
- `constexpr tuple (const tuple &)=default`
- `template<typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_ctor<const tuple<_←
UElements...>&&>(), _ImplicitCtor<_Valid, const _UElements &...> = true>
constexpr tuple (const tuple<_UElements...> &__in) noexcept(__nothrow_constructible<const _UElements
&...>())`
- `template<typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_ctor<const tuple<_←
UElements...>&&>(), _ExplicitCtor<_Valid, const _UElements &...> = false>
constexpr tuple (const tuple<_UElements...> &__in) noexcept(__nothrow_constructible<const _UElements
&...>())`
- `constexpr tuple (tuple &&)=default`
- `template<typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_ctor<tuple<_←
UElements...>&&>(), _ImplicitCtor<_Valid, _UElements...> = true>
constexpr tuple (tuple<_UElements...> &&__in) noexcept(__nothrow_constructible<_UElements...>())`
- `template<typename... _UElements, bool _Valid = (sizeof...(Elements) == sizeof...(UElements)) && !_use_other_ctor<tuple<_←
UElements...>&&>(), _ExplicitCtor<_Valid, _UElements...> = false>
constexpr tuple (tuple<_UElements...> &&__in) noexcept(__nothrow_constructible<_UElements...>())`

- constexpr **tuple** & **operator=** (__conditional_t< __assignable< _Elements... >(), **tuple** &&, __nonesuch && > __in) noexcept(__nothrow_assignable< _Elements... >())
- constexpr **tuple** & **operator=** (__conditional_t< __assignable< const _Elements &... >(), const **tuple** &, const __nonesuch & > __in) noexcept(__nothrow_assignable< const _Elements &... >())
- template<typename... _UElements>
constexpr __enable_if_t< __assignable< const _UElements &... >(), **tuple** & > **operator=** (const **tuple**< _↵ _UElements... > & __in) noexcept(__nothrow_assignable< const _UElements &... >())
- template<typename... _UElements>
constexpr __enable_if_t< __assignable< _UElements... >(), **tuple** & > **operator=** (**tuple**< _UElements... > && __in) noexcept(__nothrow_assignable< _UElements... >())
- constexpr void **swap** (**tuple** & __in) noexcept(__and< __is_nothrow_swappable< _Elements >... >::value)

6.1015.1 Detailed Description

template<typename... _Elements>

class std::tuple< _Elements >

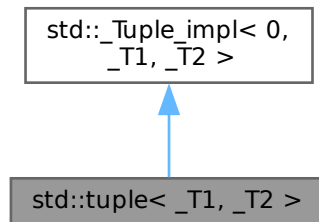
Primary class template, tuple.

The documentation for this class was generated from the following file:

- [tuple](#)

6.1016 std::tuple< _T1, _T2 > Class Template Reference

Inheritance diagram for std::tuple< _T1, _T2 >:



Public Member Functions

- template<typename _U1, typename _U2, _ImplicitCtor<!__is_alloc_arg< _U1 >(), _U1, _U2 > = true>
constexpr **tuple** (_U1 && __a1, _U2 && __a2) noexcept(__nothrow_constructible< _U1, _U2 >())
- template<typename _U1, typename _U2, _ExplicitCtor<!__is_alloc_arg< _U1 >(), _U1, _U2 > = false>
constexpr **tuple** (_U1 && __a1, _U2 && __a2) noexcept(__nothrow_constructible< _U1, _U2 >())
- template<typename _Alloc, _ImplicitDefaultCtor< [is_object](#)< _Alloc >::value, _T1, _T2 > = true>
constexpr **tuple** (allocator_arg_t __tag, const _Alloc & __a)
- template<typename _Alloc, typename _U1, typename _U2, _ImplicitCtor< true, _U1, _U2 > = true>
constexpr **tuple** (allocator_arg_t __tag, const _Alloc & __a, _U1 && __a1, _U2 && __a2)
- template<typename _Alloc, typename _U1, typename _U2, _ExplicitCtor< true, _U1, _U2 > = false>
constexpr **tuple** (allocator_arg_t __tag, const _Alloc & __a, _U1 && __a1, _U2 && __a2)

- Generated by Doxygen

- constexpr [tuple](#) & **operator=** (__conditional_t< __assignable< _T1, _T2 >(), [tuple](#) &&, __nonesuch && > __in) noexcept(__nothrow_assignable< _T1, _T2 >())
- constexpr [tuple](#) & **operator=** (__conditional_t< __assignable< const _T1 &, const _T2 & >(), const [tuple](#) &, const __nonesuch & > __in) noexcept(__nothrow_assignable< const _T1 &, const _T2 & >())
- template<typename _U1, typename _U2 >
constexpr __enable_if_t< __assignable< const _U1 &, const _U2 & >(), [tuple](#) & > **operator=** (const [pair](#)< _U1, _U2 > & __in) noexcept(__nothrow_assignable< const _U1 &, const _U2 & >())
- template<typename _U1, typename _U2 >
constexpr __enable_if_t< __assignable< const _U1 &, const _U2 & >(), [tuple](#) & > **operator=** (const [tuple](#)< _U1, _U2 > & __in) noexcept(__nothrow_assignable< const _U1 &, const _U2 & >())
- template<typename _U1, typename _U2 >
constexpr __enable_if_t< __assignable< _U1, _U2 >(), [tuple](#) & > **operator=** ([pair](#)< _U1, _U2 > && __in) noexcept(__nothrow_assignable< _U1, _U2 >())
- template<typename _U1, typename _U2 >
constexpr __enable_if_t< __assignable< _U1, _U2 >(), [tuple](#) & > **operator=** ([tuple](#)< _U1, _U2 > && __in) noexcept(__nothrow_assignable< _U1, _U2 >())
- constexpr void **swap** ([tuple](#) & __in) noexcept(__and< __is_nothrow_swappable< _T1 >, __is_nothrow_swappable< _T2 > >::value)

6.1016.1 Detailed Description

template<typename _T1, typename _T2>
class std::tuple< _T1, _T2 >

Partial specialization, 2-element tuple. Includes construction and assignment from a pair.
The documentation for this class was generated from the following file:

- [tuple](#)

6.1017 std::tuple_element< __i, _Tp > Struct Template Reference

6.1017.1 Detailed Description

template<size_t __i, typename _Tp>
struct std::tuple_element< __i, _Tp >

Gives the type of the ith element of a given tuple type.
The documentation for this struct was generated from the following file:

- [utility.h](#)

6.1018 std::tuple_element< 0, pair< _Tp1, _Tp2 > > Struct Template Reference

```
#include <stl_pair.h>
```

Public Types

- typedef _Tp1 type

6.1018.1 Detailed Description

template<class _Tp1, class _Tp2>
struct std::tuple_element< 0, pair< _Tp1, _Tp2 > >

Partial specialization for std::pair.
The documentation for this struct was generated from the following file:

- [stl_pair.h](#)

6.1019 `std::tuple_element< 1, pair< _Tp1, _Tp2 > >` Struct Template Reference

`#include <stl_pair.h>`

Public Types

- `typedef _Tp2 type`

6.1019.1 Detailed Description

```
template<class _Tp1, class _Tp2>
struct std::tuple_element< 1, pair< _Tp1, _Tp2 > >
```

Partial specialization for `std::pair`.

The documentation for this struct was generated from the following file:

- [stl_pair.h](#)

6.1020 `std::tuple_element< __i, tuple< _Types... > >` Struct Template Reference

Public Types

- using `type` = `typename _Nth_type< __i, _Types... >::type`

6.1020.1 Detailed Description

```
template<size_t __i, typename... _Types>
struct std::tuple_element< __i, tuple< _Types... > >
```

Trait to get the *l*th element type from a tuple.

The documentation for this struct was generated from the following file:

- [tuple](#)

6.1021 `std::tuple_element< _Ind, array< _Tp, _Nm > >` Struct Template Reference

Public Types

- using `type` = `_Tp`

6.1021.1 Detailed Description

```
template<size_t _Ind, typename _Tp, size_t _Nm>
struct std::tuple_element< _Ind, array< _Tp, _Nm > >
```

Partial specialization for `std::array`.

The documentation for this struct was generated from the following file:

- [array](#)

6.1022 `std::tuple_size< _Tp >` Struct Template Reference

Inherited by `std::tuple_size< const __enable_if_has_tuple_size< _Tp > >`, `std::tuple_size< const volatile __enable_if_has_tuple_size< _Tp > >`, and `std::tuple_size< volatile __enable_if_has_tuple_size< _Tp > >`.

6.1022.1 Detailed Description

```
template<typename _Tp>
struct std::tuple_size< _Tp >
```

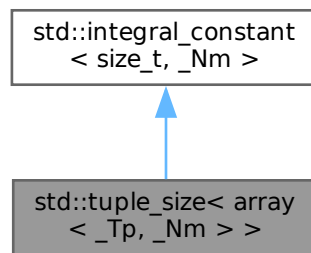
Finds the size of a given tuple type.

The documentation for this struct was generated from the following file:

- [utility.h](#)

6.1023 std::tuple_size< array< _Tp, _Nm > > Struct Template Reference

Inheritance diagram for std::tuple_size< array< _Tp, _Nm > >:



Public Types

- typedef [integral_constant](#)< size_t, __v > **type**
- typedef size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr size_t **value**

6.1023.1 Detailed Description

```
template<typename _Tp, size_t _Nm>
struct std::tuple_size< array< _Tp, _Nm > >
```

Partial specialization for std::array.

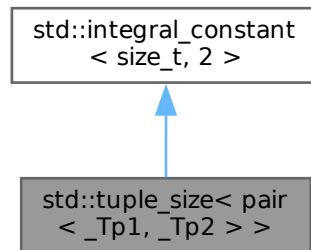
The documentation for this struct was generated from the following file:

- [array](#)

6.1024 `std::tuple_size< pair< _Tp1, _Tp2 > >` Struct Template Reference

```
#include <stl_pair.h>
```

Inheritance diagram for `std::tuple_size< pair< _Tp1, _Tp2 > >`:



Public Types

- typedef [integral_constant](#)< size_t, __v > **type**
- typedef size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr size_t **value**

6.1024.1 Detailed Description

```
template<class _Tp1, class _Tp2>
struct std::tuple_size< pair< _Tp1, _Tp2 > >
```

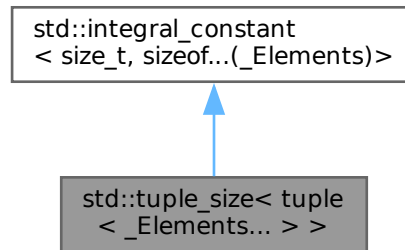
Partial specialization for `std::pair`.

The documentation for this struct was generated from the following file:

- [stl_pair.h](#)

6.1025 std::tuple_size< tuple< _Elements... > > Struct Template Reference

Inheritance diagram for std::tuple_size< tuple< _Elements... > >:

**Public Types**

- typedef [integral_constant](#)< size_t, __v > **type**
- typedef size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr size_t **value**

6.1025.1 Detailed Description

```
template<typename... _Elements>
struct std::tuple_size< tuple< _Elements... > >
```

```
class tuple_size
```

The documentation for this struct was generated from the following file:

- [tuple](#)

6.1026 __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >::type Struct Reference

```
#include <entry_cmp.hpp>
Inherits Cmp_Fn.
```

Public Member Functions

- **type** (const Cmp_Fn &other)
- bool **operator()** (entry lhs, entry rhs) const

6.1026.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >::type
```

Compare plus entry.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

6.1027 std::type_index Struct Reference

Public Member Functions

- **type_index** (const [type_info](#) &__rhs) noexcept
- **size_t hash_code** () const noexcept
- **const char * name** () const noexcept
- **bool operator!=** (const [type_index](#) &__rhs) const noexcept
- **bool operator<** (const [type_index](#) &__rhs) const noexcept
- **bool operator<=** (const [type_index](#) &__rhs) const noexcept
- **bool operator==** (const [type_index](#) &__rhs) const noexcept
- **bool operator>** (const [type_index](#) &__rhs) const noexcept
- **bool operator>=** (const [type_index](#) &__rhs) const noexcept

6.1027.1 Detailed Description

Class [type_index](#).

The class [type_index](#) provides a simple wrapper for [type_info](#) which can be used as an index type in associative containers (23.6) and in unordered associative containers (23.7).

The documentation for this struct was generated from the following file:

- [typeidindex](#)

6.1028 std::type_info Class Reference

Inherited by [__cxxabiv1::__array_type_info](#), [__cxxabiv1::__class_type_info](#), [__cxxabiv1::__enum_type_info](#), [__cxxabiv1::__function_type_info](#), [__cxxabiv1::__fundamental_type_info](#), and [__cxxabiv1::__pbase_type_info](#).

Public Member Functions

- virtual [~type_info](#) ()
- virtual **bool __do_catch** (const [type_info](#) *__thr_type, void **__thr_obj, unsigned __outer) const
- virtual **bool __do_upcast** (const [__cxxabiv1::__class_type_info](#) *__target, void **__obj_ptr) const
- virtual **bool __is_function_p** () const
- virtual **bool __is_pointer_p** () const
- **bool before** (const [type_info](#) &__arg) const noexcept
- **size_t hash_code** () const noexcept
- **const char * name** () const noexcept
- [_GLIBCXX23_CONSTEXPR](#) **bool operator==** (const [type_info](#) &__arg) const noexcept

Protected Member Functions

- **type_info** (const char *__n)

Protected Attributes

- `const char * __name`

6.1028.1 Detailed Description

Part of RTTI.

The `type_info` class describes type information generated by an implementation.

6.1028.2 Constructor & Destructor Documentation

`~type_info()`

```
virtual std::type_info::~~type_info ( ) [virtual]
```

Destructor first. Being the first non-inline virtual function, this controls in which translation unit the vtable is emitted. The compiler makes use of that information to know where to emit the runtime-mandated `type_info` structures in the new-abi.

6.1028.3 Member Function Documentation

`before()`

```
bool std::type_info::before (
    const type_info & __arg ) const [noexcept]
```

Returns true if `*this` precedes `__arg` in the implementation's collation order.

`name()`

```
const char * std::type_info::name ( ) const [inline], [noexcept]
```

Returns an *implementation-defined* byte string; this is not portable between compilers!

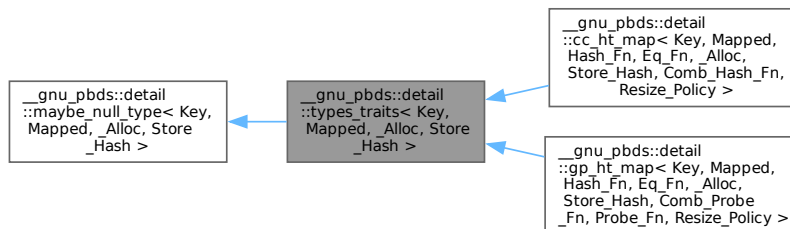
The documentation for this class was generated from the following file:

- [typeinfo](#)

6.1029 `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >` Struct Template Reference

```
#include <types_traits.hpp>
```

Inheritance diagram for `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >`:



Public Types

- typedef [std::pair](#)< size_type, size_type > **comp_hash**
- typedef __rebind_va::const_pointer **const_pointer**
- typedef __rebind_va::const_reference **const_reference**
- typedef __rebind_ka::const_pointer **key_const_pointer**
- typedef __rebind_ka::const_reference **key_const_reference**
- typedef __rebind_ka::pointer **key_pointer**
- typedef __rebind_ka::reference **key_reference**
- typedef Key **key_type**
- typedef __rebind_ma::const_pointer **mapped_const_pointer**
- typedef __rebind_ma::const_reference **mapped_const_reference**
- typedef __rebind_ma::pointer **mapped_pointer**
- typedef __rebind_ma::reference **mapped_reference**
- typedef Mapped **mapped_type**
- typedef __nothrowcopy::indicator **no_throw_indicator**
- typedef __rebind_va::pointer **pointer**
- typedef __rebind_va::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef [stored_data](#)< [value_type](#), size_type, Store_Hash > **stored_data_type**
- typedef [select_value_type](#)< Key, Mapped >::type **value_type**

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

6.1029.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >
```

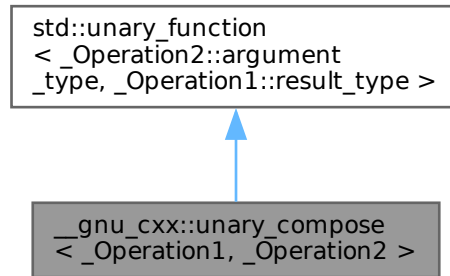
Traits for abstract types.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

6.1030 `__gnu_cxx::unary_compose< _Operation1, _Operation2 >` Class Template Reference

Inheritance diagram for `__gnu_cxx::unary_compose< _Operation1, _Operation2 >`:



Public Types

- typedef `_Operation2::argument_type` [argument_type](#)
- typedef `_Operation1::result_type` [result_type](#)

Public Member Functions

- **unary_compose** (const `_Operation1` &__x, const `_Operation2` &__y)
- `_Operation1::result_type` **operator()** (const typename `_Operation2::argument_type` &__x) const

Protected Attributes

- `_Operation1` **_M_fn1**
- `_Operation2` **_M_fn2**

6.1030.1 Detailed Description

```
template<class _Operation1, class _Operation2>
class __gnu_cxx::unary_compose< _Operation1, _Operation2 >
```

An [SGI extension](#) .

6.1030.2 Member Typedef Documentation

argument_type

```
typedef _Operation2::argument_type std::unary\_function< _Operation2::argument_type , _Operation1↔
::result_type >::argument_type [inherited]
argument_type is the type of the argument
```

result_type

```
typedef _Operation1::result_type std::unary\_function< _Operation2::argument_type , _Operation1↔
::result_type >::result_type [inherited]
```

result_type is the return type

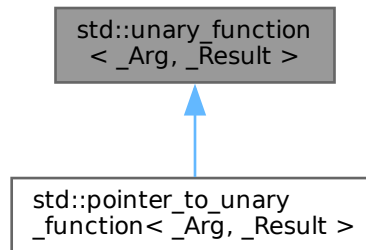
The documentation for this class was generated from the following file:

- [ext/functional](#)

6.1031 std::unary_function<_Arg, _Result> Struct Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::unary_function<_Arg, _Result>:



Public Types

- typedef _Arg [argument_type](#)
- typedef _Result [result_type](#)

6.1031.1 Detailed Description

```
template<typename _Arg, typename _Result>
struct std::unary_function<_Arg, _Result>
```

Helper for defining adaptable unary function objects.

Deprecated Deprecated in C++11, no longer in the standard since C++17.

6.1031.2 Member Typedef Documentation

argument_type

```
template<typename _Arg , typename _Result >
typedef _Arg std::unary\_function<\_Arg, \_Result>::argument\_type
argument_type is the type of the argument
```

result_type

```
template<typename _Arg , typename _Result >
typedef _Result std::unary\_function<\_Arg, \_Result>::result\_type
result_type is the return type
```

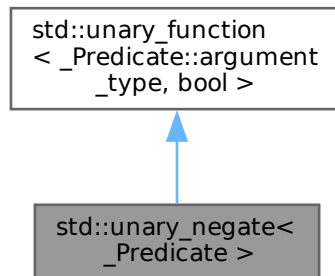
The documentation for this struct was generated from the following file:

- [stl_function.h](#)

6.1032 std::unary_negate< _Predicate > Class Template Reference

```
#include <stl_function.h>
```

Inheritance diagram for std::unary_negate< _Predicate >:

**Public Types**

- typedef `_Predicate::argument_type` [argument_type](#)
- typedef `bool` [result_type](#)

Public Member Functions

- constexpr **unary_negate** (const `_Predicate` &__x)
- constexpr `bool` **operator()** (const typename `_Predicate::argument_type` &__x) const

Protected Attributes

- `_Predicate` **_M_pred**

6.1032.1 Detailed Description

```
template<typename _Predicate>
class std::unary_negate< _Predicate >
```

One of the [negation functors](#).

6.1032.2 Member Typedef Documentation**argument_type**

```
typedef _Predicate::argument_type std::unary\_function< _Predicate::argument_type , bool >::argument_type [inherited]
```

`argument_type` is the type of the argument

result_type

```
typedef bool std::unary\_function< _Predicate::argument_type , bool >::result_type [inherited]
```

`result_type` is the return type

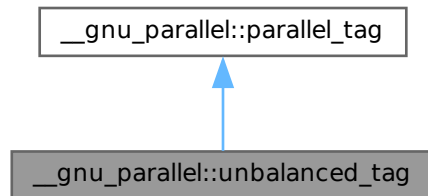
The documentation for this class was generated from the following file:

- [stl_function.h](#)

6.1033 __gnu_parallel::unbalanced_tag Struct Reference

```
#include <tags.h>
```

Inheritance diagram for __gnu_parallel::unbalanced_tag:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

6.1033.1 Detailed Description

Recommends parallel execution using static load-balancing at compile time.

6.1033.2 Member Function Documentation

`__get_num_threads()`

```
\_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ( ) [inline], [inherited]
```

Find out desired number of threads.

Returns

Desired number of threads.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::__parallel_sort\(\)](#), and [__gnu_parallel::__parallel_sort\(\)](#).

`set_num_threads()`

```
void __gnu_parallel::parallel_tag::set_num_threads (
    \_ThreadIndex __num_threads ) [inline], [inherited]
```

Set the desired number of threads.

Parameters

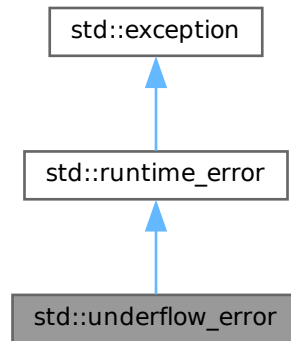
<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

The documentation for this struct was generated from the following file:

- [tags.h](#)

6.1034 `std::underflow_error` Class Reference

Inheritance diagram for `std::underflow_error`:



Public Member Functions

- `underflow_error` (const char *) `_GLIBCXX_TXN_SAFE`
- `underflow_error` (const [string](#) & __arg) `_GLIBCXX_TXN_SAFE`
- `underflow_error` (const `underflow_error` &)=default
- `underflow_error` (`underflow_error` &&)=default
- `underflow_error` & `operator=` (const `underflow_error` &)=default
- `underflow_error` & `operator=` (`underflow_error` &&)=default
- virtual const char * `what` () const noexcept

6.1034.1 Detailed Description

Thrown to indicate arithmetic underflow.

6.1034.2 Member Function Documentation

`what()`

```
virtual const char * std::runtime_error::what ( ) const [virtual], [noexcept], [inherited]
```

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::filesystem::filesystem_error](#), and [std::experimental::filesystem::v1::filesystem_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

6.1035 `std::underlying_type<_Tp>` Struct Template Reference

Inherits `__underlying_type_impl<_Tp>`.

6.1035.1 Detailed Description

```
template<typename _Tp>
struct std::underlying_type< _Tp >
```

The underlying type of an enum.

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.1036 std::uniform_int_distribution< _IntType > Class Template Reference

```
#include <uniform_int_dist.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _IntType [result_type](#)

Public Member Functions

- [uniform_int_distribution](#) ()
- [uniform_int_distribution](#) (_IntType __a, _IntType __b= [__gnu_cxx::__int_traits](#)< _IntType >::__max)
- [uniform_int_distribution](#) (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomBitGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomBitGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomBitGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomBitGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomBitGenerator >
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomBitGenerator &__urng, const [param_type](#) &__p)
- [result_type a](#) () const
- [result_type b](#) () const
- [result_type max](#) () const
- [result_type min](#) () const
- template<typename _UniformRandomBitGenerator >
[result_type operator\(\)](#) (_UniformRandomBitGenerator &__urng)
- template<typename _UniformRandomBitGenerator >
[result_type operator\(\)](#) (_UniformRandomBitGenerator &__urng, const [param_type](#) &__p)
- [param_type param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- bool [operator==](#) (const [uniform_int_distribution](#) &__d1, const [uniform_int_distribution](#) &__d2)

6.1036.1 Detailed Description

```
template<typename _IntType = int>
class std::uniform_int_distribution< _IntType >
```

Uniform discrete distribution for random numbers. A discrete random distribution on the range $[min, max]$ with equal probability throughout the range.

6.1036.2 Member Typedef Documentation

result_type

```
template<typename _IntType = int>
typedef _IntType std::uniform_int_distribution< _IntType >::result_type
```

The type of the range of the distribution.

6.1036.3 Constructor & Destructor Documentation

uniform_int_distribution() [1/2]

```
template<typename _IntType = int>
std::uniform_int_distribution< _IntType >::uniform_int_distribution ( ) [inline]
```

Constructs a uniform distribution object.

uniform_int_distribution() [2/2]

```
template<typename _IntType = int>
std::uniform_int_distribution< _IntType >::uniform_int_distribution (
    _IntType __a,
    _IntType __b = __gnu_cxx::__int_traits<_IntType>::__max ) [inline], [explicit]
```

Constructs a uniform distribution object.

6.1036.4 Member Function Documentation

max()

```
template<typename _IntType = int>
result_type std::uniform_int_distribution< _IntType >::max ( ) const [inline]
```

Returns the inclusive upper bound of the distribution range.

min()

```
template<typename _IntType = int>
result_type std::uniform_int_distribution< _IntType >::min ( ) const [inline]
```

Returns the inclusive lower bound of the distribution range.

operator>()()

```
template<typename _IntType = int>
template<typename _UniformRandomBitGenerator >
result_type std::uniform_int_distribution< _IntType >::operator() (
    _UniformRandomBitGenerator & __urng ) [inline]
```

Generating functions.

References `std::uniform_int_distribution<_IntType>::operator>()()`.

Referenced by `std::uniform_int_distribution<_IntType>::operator>()()`.

param() [1/2]

```
template<typename _IntType = int>
param_type std::uniform_int_distribution< _IntType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.
Referenced by [std::operator>>\(\)](#).

param() [2/2]

```
template<typename _IntType = int>
void std::uniform_int_distribution< _IntType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _IntType = int>
void std::uniform_int_distribution< _IntType >::reset ( ) [inline]
```

Resets the distribution state.
Does nothing for the uniform integer distribution.

6.1036.5 Friends And Related Symbol Documentation**operator==**

```
template<typename _IntType = int>
bool operator== (
    const uniform_int_distribution< _IntType > & __d1,
    const uniform_int_distribution< _IntType > & __d2 ) [friend]
```

Return true if two uniform integer distributions have the same parameters.

The documentation for this class was generated from the following file:

- [uniform_int_dist.h](#)

6.1037 std::uniform_real_distribution< _RealType > Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- [uniform_real_distribution](#) ()
- [uniform_real_distribution](#) (`_RealType` __a, `_RealType` __b= `_RealType`(1))
- [uniform_real_distribution](#) (const [param_type](#) &__p)

- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator>`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator>`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `template<typename _UniformRandomNumberGenerator>`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `result_type a () const`
- `result_type b () const`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `param_type param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `bool operator== (const uniform_real_distribution &__d1, const uniform_real_distribution &__d2)`

6.1037.1 Detailed Description

`template<typename _RealType = double>`
class `std::uniform_real_distribution<_RealType>`

Uniform continuous distribution for random numbers.

A continuous random distribution on the range [min, max) with equal probability throughout the range. The URNG should be real-valued and deliver number in the range [0, 1).

6.1037.2 Member Typedef Documentation

result_type

`template<typename _RealType = double>`
`typedef _RealType std::uniform_real_distribution<_RealType>::result_type`
 The type of the range of the distribution.

6.1037.3 Constructor & Destructor Documentation

uniform_real_distribution() [1/2]

`template<typename _RealType = double>`
`std::uniform_real_distribution<_RealType>::uniform_real_distribution () [inline]`
 Constructs a `uniform_real_distribution` object.
 The lower bound is set to 0.0 and the upper bound to 1.0

uniform_real_distribution() [2/2]

`template<typename _RealType = double>`
`std::uniform_real_distribution<_RealType>::uniform_real_distribution (`
`_RealType __a,`
`_RealType __b = _RealType(1)) [inline], [explicit]`

Constructs a `uniform_real_distribution` object.

Parameters

<code>_↔ _a</code>	[IN] The lower bound of the distribution.
<code>_↔ _b</code>	[IN] The upper bound of the distribution.

6.1037.4 Member Function Documentation

`max()`

```
template<typename _RealType = double>
result_type std::uniform_real_distribution< _RealType >::max ( ) const [inline]
```

Returns the inclusive upper bound of the distribution range.

`min()`

```
template<typename _RealType = double>
result_type std::uniform_real_distribution< _RealType >::min ( ) const [inline]
```

Returns the inclusive lower bound of the distribution range.

`operator>()`

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::uniform_real_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References [std::uniform_real_distribution< _RealType >::operator>\(\)](#).

Referenced by [std::uniform_real_distribution< _RealType >::operator>\(\)](#).

`param()` [1/2]

```
template<typename _RealType = double>
param_type std::uniform_real_distribution< _RealType >::param ( ) const [inline]
```

Returns the parameter set of the distribution.

Referenced by [std::operator>>\(\)](#).

`param()` [2/2]

```
template<typename _RealType = double>
void std::uniform_real_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::uniform_real_distribution<_RealType>::reset ( ) [inline]
```

Resets the distribution state.

Does nothing for the uniform real distribution.

6.1037.5 Friends And Related Symbol Documentation**operator==**

```
template<typename _RealType = double>
bool operator== (
    const uniform_real_distribution<_RealType> & __d1,
    const uniform_real_distribution<_RealType> & __d2 ) [friend]
```

Return true if two uniform real distributions have the same parameters.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.1038 std::unique_lock<_Mutex> Class Template Reference

```
#include <unique_lock.h>
```

Public Types

- typedef _Mutex **mutex_type**

Public Member Functions

- **unique_lock** (const [unique_lock](#) &)=delete
- **unique_lock** (mutex_type &__m)
- **unique_lock** (mutex_type &__m, [adopt_lock_t](#)) noexcept
- template<typename _Rep, typename _Period>
 unique_lock (mutex_type &__m, const [chrono::duration](#)<_Rep, _Period> &__rtime)
- template<typename _Clock, typename _Duration>
 unique_lock (mutex_type &__m, const [chrono::time_point](#)<_Clock, _Duration> &__atime)
- **unique_lock** (mutex_type &__m, [defer_lock_t](#)) noexcept
- **unique_lock** (mutex_type &__m, [try_to_lock_t](#))
- **unique_lock** ([unique_lock](#) &&__u) noexcept
- void **lock** ()
- mutex_type * **mutex** () const noexcept
- **operator bool** () const noexcept
- [unique_lock](#) & **operator=** (const [unique_lock](#) &)=delete
- [unique_lock](#) & **operator=** ([unique_lock](#) &&__u) noexcept
- bool **owns_lock** () const noexcept
- mutex_type * **release** () noexcept
- void **swap** ([unique_lock](#) &__u) noexcept
- bool **try_lock** ()
- template<typename _Rep, typename _Period>
 bool **try_lock_for** (const [chrono::duration](#)<_Rep, _Period> &__rtime)
- template<typename _Clock, typename _Duration>
 bool **try_lock_until** (const [chrono::time_point](#)<_Clock, _Duration> &__atime)
- void **unlock** ()

Related Symbols

(Note that these are not member symbols.)

- `template<typename _Mutex >`
`void swap (unique_lock< _Mutex > &__x, unique_lock< _Mutex > &__y) noexcept`

6.1038.1 Detailed Description

template<typename _Mutex>
class std::unique_lock< _Mutex >

A movable scoped lock type.

A `unique_lock` controls mutex ownership within a scope. Ownership of the mutex can be delayed until after construction and can be transferred to another `unique_lock` by move construction or move assignment. If a mutex lock is owned when the destructor runs ownership will be released.

6.1038.2 Friends And Related Symbol Documentation

swap()

```
template<typename _Mutex >
void swap (
    unique_lock< _Mutex > & __x,
    unique_lock< _Mutex > & __y ) [related]
```

Swap overload for `unique_lock` objects.

The documentation for this class was generated from the following file:

- [unique_lock.h](#)

6.1039 std::unique_ptr< _Tp, _Dp > Class Template Reference

```
#include <memory>
```

Public Types

- using **deleter_type** = `_Dp`
- using **element_type** = `_Tp`
- using **pointer** = `typename __uniq_ptr_impl< _Tp, _Dp >::pointer`

Public Member Functions

- `template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>`
`constexpr unique_ptr () noexcept`
- `template<typename _Up, typename = _Require< is_convertible<_Up*, _Tp*>, is_same<_Dp, default_delete<_Tp>>>>`
`unique_ptr (auto_ptr< _Up > &&__u) noexcept`
- `unique_ptr (const unique_ptr &)=delete`
- `template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>`
`constexpr unique_ptr (nullptr_t) noexcept`
- `template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>`
`_GLIBCXX23_CONSTEXPR unique_ptr (pointer __p) noexcept`
- `template<typename _Del = deleter_type, typename = _Require<is_move_constructible<_Del>>>`
`_GLIBCXX23_CONSTEXPR unique_ptr (pointer __p, __enable_if_t<!is_lvalue_reference< _Del >::value, _Del && > __d) noexcept`
- `template<typename _Del = deleter_type, typename = _Require<is_copy_constructible<_Del>>>`
`_GLIBCXX23_CONSTEXPR unique_ptr (pointer __p, const deleter_type &__d) noexcept`

- `template<typename _Del = deleter_type, typename _DelUnref = typename remove_reference<_Del>::type>
_GLIBCXX23_CONSTEXPR unique_ptr (pointer, __enable_if_t< is_lvalue_reference< _Del >::value, _Del↔
Unref && >)=delete`
- `unique_ptr (unique_ptr &&)=default`
- `template<typename _Up, typename _Ep, typename = _Require< __safe_conversion_up<_Up, _Ep>, __conditional_t<is_reference<↔
_Dp>::value, is_same< _Ep, _Dp>, is_convertible< _Ep, _Dp>>>>
_GLIBCXX23_CONSTEXPR unique_ptr (unique_ptr< _Up, _Ep > &&__u) noexcept`
- `~unique_ptr () noexcept`
- `_GLIBCXX23_CONSTEXPR pointer get () const noexcept`
- `_GLIBCXX23_CONSTEXPR const deleter_type & get_deleter () const noexcept`
- `_GLIBCXX23_CONSTEXPR deleter_type & get_deleter () noexcept`
- `_GLIBCXX23_CONSTEXPR operator bool () const noexcept`
- `_GLIBCXX23_CONSTEXPR add_lvalue_reference< element_type >::type operator* () const noexcept(noexcept(*std::declval<
pointer >()))`
- `_GLIBCXX23_CONSTEXPR pointer operator-> () const noexcept`
- `unique_ptr & operator= (const unique_ptr &)=delete`
- `_GLIBCXX23_CONSTEXPR unique_ptr & operator= (nullptr_t) noexcept`
- `unique_ptr & operator= (unique_ptr &&)=default`
- `template<typename _Up, typename _Ep >
_GLIBCXX23_CONSTEXPR enable_if< __and< __safe_conversion_up< _Up, _Ep >, is_assignable<
deleter_type &, _Ep && >::value, unique_ptr & >::type operator= (unique_ptr< _Up, _Ep > &&__u) noexcept`
- `_GLIBCXX23_CONSTEXPR pointer release () noexcept`
- `_GLIBCXX23_CONSTEXPR void reset (pointer __p=pointer()) noexcept`
- `_GLIBCXX23_CONSTEXPR void swap (unique_ptr &__u) noexcept`

Related Symbols

(Note that these are not member symbols.)

- `template<typename _Tp, typename... _Args>
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique (_Args &&... __args)`
- `template<typename _Tp, typename... _Args>
__detail::__invalid_make_unique_t< _Tp > make_unique (_Args &&...)=delete`
- `template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t< _Tp > make_unique (size_t __num)`
- `template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_t< _Tp > make_unique_for_overwrite ()`
- `template<typename _Tp, typename... _Args>
__detail::__invalid_make_unique_t< _Tp > make_unique_for_overwrite (_Args &&...)=delete`
- `template<typename _Tp >
_GLIBCXX23_CONSTEXPR __detail::__unique_ptr_array_t< _Tp > make_unique_for_overwrite (size_t __num)`
- `template<typename _CharT, typename _Traits, typename _Tp, typename _Dp >
basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const unique_ptr<
_Tp, _Dp > &__p)`
- `template<typename _Tp, typename _Dp >
_GLIBCXX23_CONSTEXPR enable_if< __is_swappable< _Dp >::value >::type swap (unique_ptr< _Tp, _Dp
> &__x, unique_ptr< _Tp, _Dp > &__y) noexcept`

6.1039.1 Detailed Description

```
template<typename _Tp, typename _Dp = default_delete<_Tp>>
class std::unique_ptr< _Tp, _Dp >
```

A move-only smart pointer that manages unique ownership of a resource.

Since

C++11

6.1039.2 Constructor & Destructor Documentation

unique_ptr() [1/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr std::unique_ptr< _Tp, _Dp >::unique_ptr ( ) [inline], [constexpr], [noexcept]
```

Default constructor, creates a unique_ptr that owns nothing.

unique_ptr() [2/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp, _Dp >::unique_ptr (
    pointer __p ) [inline], [explicit], [noexcept]
```

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an object of element_type
-------------------------	--

The deleter will be value-initialized.

unique_ptr() [3/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Del = deleter_type, typename = _Require<is_copy_constructible<_Del>>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp, _Dp >::unique_ptr (
    pointer __p,
    const deleter_type & __d ) [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an object of element_type
\leftrightarrow _d	A reference to a deleter.

The deleter will be initialized with __d

unique_ptr() [4/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Del = deleter_type, typename = _Require<is_move_constructible<_Del>>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp, _Dp >::unique_ptr (
    pointer __p,
    __enable_if_t<!is_lvalue_reference< _Del >::value, _Del && > __d ) [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an object of element_type
\leftrightarrow _d	An rvalue reference to a (non-reference) deleter.

The deleter will be initialized with `std::move(__d)`

unique_ptr() [5/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr std::unique_ptr<_Tp, _Dp>::unique_ptr (
    nullptr_t ) [inline], [constexpr], [noexcept]
```

Creates a unique_ptr that owns nothing.

unique_ptr() [6/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
std::unique_ptr<_Tp, _Dp>::unique_ptr (
    unique_ptr<_Tp, _Dp> && ) [default]
```

Move constructor.

unique_ptr() [7/8]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Up , typename _Ep , typename = _Require< __safe_conversion_up<_Up, _Ep>, __
conditional_t<is_reference<_Dp>::value, is_same<_Ep, _Dp>, is_convertible<_Ep, _Dp>>>>
_GLIBCXX23_CONSTEXPR std::unique_ptr<_Tp, _Dp>::unique_ptr (
    unique_ptr<_Up, _Ep> && __u ) [inline], [noexcept]
```

Converting constructor from another type.

Requires that the pointer owned by __u is convertible to the type of pointer owned by this object, __u does not own an array, and __u has a compatible deleter type.

unique_ptr() [8/8]

```
template<typename _Tp , typename _Dp >
template<typename _Up , typename >
std::unique_ptr<_Tp, _Dp>::unique_ptr (
    auto_ptr<_Up> && __u ) [inline], [noexcept]
```

Converting constructor from auto_ptr.

~unique_ptr()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
std::unique_ptr<_Tp, _Dp>::~~unique_ptr ( ) [inline], [noexcept]
```

Destructor, invokes the deleter if the stored pointer is not null.

References `std::unique_ptr<_Tp, _Dp>::get_deleter()`, and `std::move()`.

6.1039.3 Member Function Documentation**get()**

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR pointer std::unique_ptr<_Tp, _Dp>::get ( ) const [inline], [noexcept]
```

Return the stored pointer.

Referenced by [std::unique_ptr<_Tp, _Dp>::make_unique_for_overwrite\(\)](#), [std::unique_ptr<_Tp, _Dp>::operator bool\(\)](#), [std::unique_ptr<_Tp\[\], _Dp>::operator bool\(\)](#), [std::operator!=\(\)](#), [std::unique_ptr<_Tp, _Dp>::operator*\(\)](#), [std::unique_ptr<_Tp, _Dp>::operator<\(\)](#), [std::operator<\(\)](#), [std::operator<\(\)](#), [std::operator==\(\)](#), [std::operator>\(\)](#), [std::operator>\(\)](#), and [std::unique_ptr<_Tp\[\], _Dp>::operator\[\]\(\)](#).

get_deleter() [1/2]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR const deleter_type & std::unique_ptr< _Tp, _Dp >::get_deleter ( ) const
[inline], [noexcept]
```

Return a reference to the stored deleter.

get_deleter() [2/2]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR deleter_type & std::unique_ptr< _Tp, _Dp >::get_deleter ( ) [inline], [noexcept]
```

Return a reference to the stored deleter.

Referenced by [std::unique_ptr<_Tp\[\], _Dp>::~~unique_ptr\(\)](#), [std::unique_ptr<_Tp, _Dp>::~~unique_ptr\(\)](#), [std::unique_ptr<_Tp, _Dp>::operator*\(\)](#), and [std::unique_ptr<_Tp\[\], _Dp>::operator=\(\)](#).

operator bool()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp, _Dp >::operator bool ( ) const [inline], [explicit],
[noexcept]
```

Return true if the stored pointer is not null.

References [std::unique_ptr<_Tp, _Dp>::get\(\)](#).

operator*()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR add_lvalue_reference< element_type >::type std::unique_ptr< _Tp, _Dp >↔
::operator* ( ) const [inline], [noexcept]
```

Dereference the stored pointer.

References [std::unique_ptr<_Tp, _Dp>::get\(\)](#).

operator->()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR pointer std::unique_ptr< _Tp, _Dp >::operator-> ( ) const [inline], [noexcept]
```

Return the stored pointer.

References [std::unique_ptr<_Tp, _Dp>::get\(\)](#).

operator=() [1/3]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR unique_ptr & std::unique_ptr< _Tp, _Dp >::operator= (
    nullptr_t ) [inline], [noexcept]
```

Reset the unique_ptr to empty, invoking the deleter if necessary.

References [std::unique_ptr<_Tp, _Dp>::reset\(\)](#).

operator=() [2/3]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
unique_ptr & std::unique_ptr<_Tp, _Dp >::operator= (
    unique_ptr<_Tp, _Dp > && ) [default]
```

Move assignment operator.

Invokes the deleter if this object owns a pointer.

operator=() [3/3]

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
template<typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR enable_if< __and< __safe_conversion_up< _Up, _Ep >, is_assignable< deleter<_Tp, _Dp>, deleter<_Up, _Ep> >>::value, unique_ptr & >::type std::unique_ptr<_Tp, _Dp >::operator= (
    unique_ptr<_Up, _Ep > && __u ) [inline], [noexcept]
```

Assignment from another type.

Parameters

<code>__u</code>	The object to transfer ownership from, which owns a convertible pointer to a non-array object.
------------------	--

Invokes the deleter if this object owns a pointer.

References [std::unique_ptr<_Tp, _Dp >::get_deleter\(\)](#), and [std::unique_ptr<_Tp, _Dp >::reset\(\)](#).

release()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR pointer std::unique_ptr<_Tp, _Dp >::release ( ) [inline], [noexcept]
```

Release ownership of any stored pointer.

reset()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR void std::unique_ptr<_Tp, _Dp >::reset (
    pointer __p = pointer() ) [inline], [noexcept]
```

Replace the stored pointer.

Parameters

<code>__p</code>	The new pointer to store.
------------------	---------------------------

The deleter will be invoked if a pointer is already owned.

References [std::move\(\)](#).

Referenced by [std::unique_ptr<_Tp, _Dp >::operator=\(\)](#), [std::unique_ptr<_Tp\[\], _Dp >::operator=\(\)](#), [std::unique_ptr<_Tp, _Dp >::operator=\(\)](#), and [std::unique_ptr<_Tp\[\], _Dp >::operator=\(\)](#).

swap()

```
template<typename _Tp , typename _Dp = default_delete<_Tp>>
_GLIBCXX23_CONSTEXPR void std::unique_ptr<_Tp, _Dp >::swap (
    unique_ptr<_Tp, _Dp > & __u ) [inline], [noexcept]
```

Exchange the pointer and deleter with another object.

The documentation for this class was generated from the following files:

- [unique_ptr.h](#)
- [auto_ptr.h](#)

6.1040 std::unique_ptr< _Tp[], _Dp > Class Template Reference

```
#include <memory>
```

Public Types

- template<typename _Up >
using **__safe_conversion_raw** = __and_< __or_< __or_< [is_same](#)< _Up, pointer >, [is_same](#)< _Up, nullptr_<_t > >, __and_< [is_pointer](#)< _Up >, [is_same](#)< pointer, element_type * >, [is_convertible](#)< typename [remove_pointer](#)< _Up >::type(*)[], element_type(*)[]> > > >
- template<typename _Up, typename _Ep, typename _UPtr = unique_ptr<_Up, _Ep>, typename _UP_pointer = typename _UPtr::pointer, typename _UP_element_type = typename _UPtr::element_type>
using **__safe_conversion_up** = __and_< [is_array](#)< _Up >, [is_same](#)< pointer, element_type * >, [is_same](#)< _UP_pointer, _UP_element_type * >, [is_convertible](#)< _UP_element_type(*)[], element_type(*)[]> >
- using **deleter_type** = _Dp
- using **element_type** = _Tp
- using **pointer** = typename __uniq_ptr_impl< _Tp, _Dp >::pointer

Public Member Functions

- template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr [unique_ptr](#) () noexcept
- template<typename _Up, typename _Vp = _Dp, typename = _DeleterConstraint<_Vp>, typename = typename enable_if< __safe_<__conversion_raw<_Up>::value, bool>::type>
[_GLIBCXX23_CONSTEXPR](#) [unique_ptr](#) (_Up __p) noexcept
- template<typename _Up, typename _Del = deleter_type, typename = _Require<__safe_conversion_raw<_Up>, is_move_<__constructible<_Del>>>
[_GLIBCXX23_CONSTEXPR](#) [unique_ptr](#) (_Up __p, __enable_if_t<[is_lvalue_reference](#)<_Del>::value, _Del && > __d) noexcept
- template<typename _Up, typename _Del = deleter_type, typename = _Require<__safe_conversion_raw<_Up>, is_copy_<__constructible<_Del>>>
[_GLIBCXX23_CONSTEXPR](#) [unique_ptr](#) (_Up __p, const deleter_type & __d) noexcept
- template<typename _Up, typename _Del = deleter_type, typename _DelUnref = typename remove_reference<_Del>::type, typename = _Require<__safe_conversion_raw<_Up>>>
[unique_ptr](#) (_Up, __enable_if_t< [is_lvalue_reference](#)<_Del>::value, _DelUnref && >)=delete
- [unique_ptr](#) (const [unique_ptr](#) &)=delete
- template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr [unique_ptr](#) (nullptr_t) noexcept
- [unique_ptr](#) ([unique_ptr](#) &&)=default
- template<typename _Up, typename _Ep, typename = _Require< __safe_conversion_up<_Up, _Ep>, __conditional_t<is_reference<__<_Dp>::value, is_same<_Ep, _Dp>, is_convertible<_Ep, _Dp>>>>
[_GLIBCXX23_CONSTEXPR](#) [unique_ptr](#) ([unique_ptr](#)<_Up, _Ep > && __u) noexcept
- [~unique_ptr](#) ()
- [_GLIBCXX23_CONSTEXPR](#) pointer [get](#) () const noexcept
- [_GLIBCXX23_CONSTEXPR](#) const deleter_type & [get_deleter](#) () const noexcept
- [_GLIBCXX23_CONSTEXPR](#) deleter_type & [get_deleter](#) () noexcept
- [_GLIBCXX23_CONSTEXPR](#) [operator bool](#) () const noexcept
- [unique_ptr](#) & [operator=](#) (const [unique_ptr](#) &)=delete
- [_GLIBCXX23_CONSTEXPR](#) [unique_ptr](#) & [operator=](#) (nullptr_t) noexcept
- [unique_ptr](#) & [operator=](#) ([unique_ptr](#) &&)=default

- template<typename _Up, typename _Ep >
_GLIBCXX23_CONSTEXPR enable_if< __and< __safe_conversion_up< _Up, _Ep >, is_assignable< deleter_type &, _Ep && >>::value, unique_ptr & >::type operator= (unique_ptr< _Up, _Ep > &&__u) noexcept
- _GLIBCXX23_CONSTEXPR std::add_lvalue_reference< element_type >::type operator[] (size_t __i) const
- _GLIBCXX23_CONSTEXPR pointer release () noexcept
- template<typename _Up, typename = _Require< __or<is_same<_Up, pointer>, __and<is_same<pointer, element_type*>, is_<← pointer<_Up>, is_convertible< typename remove_pointer<_Up>::type(*)[], element_type(*)[] >> >>>
_GLIBCXX23_CONSTEXPR void reset (_Up __p) noexcept
- _GLIBCXX23_CONSTEXPR void reset (nullptr_t=nullptr) noexcept
- _GLIBCXX23_CONSTEXPR void swap (unique_ptr &__u) noexcept

6.1040.1 Detailed Description

template<typename _Tp, typename _Dp>
class std::unique_ptr< _Tp[], _Dp >

A move-only smart pointer that manages unique ownership of an array.

Since

C++11

6.1040.2 Constructor & Destructor Documentation

unique_ptr() [1/6]

```
template<typename _Tp, typename _Dp >
template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr std::unique_ptr< _Tp[], _Dp >::unique_ptr ( ) [inline], [constexpr], [noexcept]
```


Default constructor, creates a unique_ptr that owns nothing.

unique_ptr() [2/6]

```
template<typename _Tp, typename _Dp >
template<typename _Up, typename _Vp = _Dp, typename = _DeleterConstraint<_Vp>, typename = typename
enable_if< __safe_conversion_raw<_Up>::value, bool>::type>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp[], _Dp >::unique_ptr (
    _Up __p ) [inline], [explicit], [noexcept]
```

Takes ownership of a pointer.

Parameters

	A pointer to an array of a type safely convertible to an array of element_type
<u>p</u>	



The deleter will be value-initialized.

unique_ptr() [3/6]

```
template<typename _Tp, typename _Dp >
template<typename _Up, typename _Del = deleter_type, typename = _Require<__safe_conversion_<←
raw<_Up>, is_copy_constructible<_Del>>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp[], _Dp >::unique_ptr (
    _Up __p,
    const deleter_type & __d ) [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

 <code>_p</code>	A pointer to an array of a type safely convertible to an array of <code>element_type</code>
 <code>_d</code>	A reference to a deleter.



The deleter will be initialized with `__d`

unique_ptr() [4/6]

```
template<typename _Tp , typename _Dp >
template<typename _Up , typename _Del = deleter_type, typename = _Require<__safe_conversion_
raw<_Up>, is_move_constructible<_Del>>>
_GLIBCXX23_CONSTEXPR std::unique_ptr< _Tp[], _Dp >::unique_ptr (
    _Up __p,
    __enable_if_t<!is_lvalue_reference< _Del >::value, _Del && > __d ) [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

 <code>_p</code>	A pointer to an array of a type safely convertible to an array of <code>element_type</code>
 <code>_d</code>	A reference to a deleter.

The deleter will be initialized with `std::move(__d)`

unique_ptr() [5/6]

```
template<typename _Tp , typename _Dp >
std::unique_ptr< _Tp[], _Dp >::unique_ptr (
    std::unique_ptr< _Tp[], _Dp > && ) [default]
```

Move constructor.

unique_ptr() [6/6]

```
template<typename _Tp , typename _Dp >
template<typename _Del = _Dp, typename = _DeleterConstraint<_Del>>
constexpr std::unique_ptr< _Tp[], _Dp >::unique_ptr (
    nullptr_t ) [inline], [constexpr], [noexcept]
```

Creates a `unique_ptr` that owns nothing.

~unique_ptr()

```
template<typename _Tp , typename _Dp >
std::unique_ptr< _Tp[], _Dp >::~~unique_ptr ( ) [inline]
```

Destructor, invokes the deleter if the stored pointer is not null.
References `std::unique_ptr< _Tp, _Dp >::get_deleter()`.

6.1040.3 Member Function Documentation**get()**

```
template<typename _Tp , typename _Dp >
```

`_GLIBCXX23_CONSTEXPR` pointer `std::unique_ptr<_Tp[], _Dp>::get () const` [inline], [noexcept]
Return the stored pointer.

get_deleter() [1/2]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR const deleter_type & std::unique_ptr<_Tp[], _Dp>::get_deleter ( ) const
[inline], [noexcept]
```

Return a reference to the stored deleter.

get_deleter() [2/2]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR deleter_type & std::unique_ptr<_Tp[], _Dp>::get_deleter ( ) [inline],
[noexcept]
```

Return a reference to the stored deleter.

operator bool()

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR std::unique_ptr<_Tp[], _Dp>::operator bool ( ) const [inline], [explicit],
[noexcept]
```

Return `true` if the stored pointer is not null.

References `std::unique_ptr<_Tp, _Dp>::get()`.

operator=() [1/3]

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR unique_ptr & std::unique_ptr<_Tp[], _Dp>::operator= (
    nullptr_t ) [inline], [noexcept]
```

Reset the `unique_ptr` to empty, invoking the deleter if necessary.

References `std::unique_ptr<_Tp, _Dp>::reset()`.

operator=() [2/3]

```
template<typename _Tp , typename _Dp >
unique_ptr & std::unique_ptr<_Tp[], _Dp>::operator= (
    unique_ptr<_Tp[], _Dp> && ) [default]
```

Move assignment operator.

Invokes the deleter if this object owns a pointer.

operator=() [3/3]

```
template<typename _Tp , typename _Dp >
template<typename _Up , typename _Ep >
_GLIBCXX23_CONSTEXPR enable_if< __and< __safe_conversion_up<_Up, _Ep>, is_assignable< deleter←
_type &, _Ep && > >::value, unique_ptr & >::type std::unique_ptr<_Tp[], _Dp>::operator= (
    unique_ptr<_Up, _Ep> && __u ) [inline], [noexcept]
```

Assignment from another type.

Parameters

<code>←_</code>	The object to transfer ownership from, which owns a convertible pointer to an array object.
<code>_U</code>	

Invokes the deleter if this object owns a pointer.

References [std::unique_ptr<_Tp, _Dp>::get_deleter\(\)](#), and [std::unique_ptr<_Tp, _Dp>::reset\(\)](#).

operator[]()

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR std::add\_lvalue\_reference< element_type >::type std::unique\_ptr< _Tp[], _Dp
>::operator[] (
    size_t __i ) const [inline]
```

Access an element of owned array.

References [std::unique_ptr<_Tp, _Dp>::get\(\)](#).

release()

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR pointer std::unique\_ptr< _Tp[], _Dp >::release ( ) [inline], [noexcept]
```

Release ownership of any stored pointer.

reset()

```
template<typename _Tp , typename _Dp >
template<typename _Up , typename = _Require< __or_<is_same<_Up, pointer>, __and_<is_same<pointer,
element_type*>, is_pointer<_Up>, is_convertible< typename remove_pointer<_Up>::type(*)[], element↔
_type(*)[] > > > >>
_GLIBCXX23_CONSTEXPR void std::unique\_ptr< _Tp[], _Dp >::reset (
    _Up __p ) [inline], [noexcept]
```

Replace the stored pointer.

Parameters

_↔	The new pointer to store.
_p	

The deleter will be invoked if a pointer is already owned.

References [std::move\(\)](#).

swap()

```
template<typename _Tp , typename _Dp >
_GLIBCXX23_CONSTEXPR void std::unique\_ptr< _Tp[], _Dp >::swap (
    unique\_ptr< _Tp[], _Dp > & __u ) [inline], [noexcept]
```

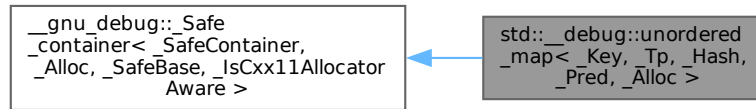
Exchange the pointer and deleter with another object.

The documentation for this class was generated from the following file:

- [unique_ptr.h](#)

6.1041 std::__debug::unordered_map<_Key,_Tp,_Hash,_Pred,_Alloc> Class Template Reference

Inheritance diagram for std::__debug::unordered_map<_Key,_Tp,_Hash,_Pred,_Alloc>:



Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef __gnu_debug::_Safe_iterator<_Base_const_iterator, unordered_map> **const_iterator**
- typedef __gnu_debug::_Safe_local_iterator<_Base_const_local_iterator, unordered_map> **const_local_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef _Base::difference_type **difference_type**
- typedef _Base::hasher **hasher**
- using **insert_return_type** = _Node_insert_return<iterator, node_type>
- typedef __gnu_debug::_Safe_iterator<_Base_iterator, unordered_map> **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef __gnu_debug::_Safe_local_iterator<_Base_local_iterator, unordered_map> **local_iterator**
- typedef _Base::mapped_type **mapped_type**
- using **node_type** = typename _Base::node_type
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_map** (_Base_ref __x)
- template<typename _InputIterator>
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator>
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator>
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (const allocator_type &__a)
- **unordered_map** (const unordered_map &)=default
- **unordered_map** (const unordered_map &__umap, const allocator_type &__a)
- **unordered_map** (initializer_list<value_type> __l, size_type __n, const allocator_type &__a)

- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (size_type __n, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** ([unordered_map](#) &&)=default
- **unordered_map** ([unordered_map](#) && __umap, const allocator_type &__a) noexcept(noexcept([_Base](#)(std::move(__umap), __a)))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) **begin** () const noexcept
- [iterator](#) **begin** () noexcept
- [local_iterator](#) **begin** (size_type __b)
- [const_local_iterator](#) **begin** (size_type __b) const
- size_type **bucket_size** (size_type __b) const
- [const_iterator](#) **cbegin** () const noexcept
- [const_local_iterator](#) **cbegin** (size_type __b) const
- [const_iterator](#) **cend** () const noexcept
- [const_local_iterator](#) **cend** (size_type __b) const
- void **clear** () noexcept
- template<typename... _Args>
[std::pair](#)< [iterator](#), bool > **emplace** (_Args &&... __args)
- template<typename... _Args>
[iterator](#) **emplace_hint** (const [iterator](#) __hint, _Args &&... __args)
- [const_iterator](#) **end** () const noexcept
- [iterator](#) **end** () noexcept
- [local_iterator](#) **end** (size_type __b)
- [const_local_iterator](#) **end** (size_type __b) const
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const _Kt &__k) const
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const key_type &__key)
- [std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const key_type &__key) const
- [_Base_iterator](#) **erase** ([_Base_const_iterator](#) __it)
- size_type **erase** (const key_type &__key)
- [iterator](#) **erase** ([const_iterator](#) __first, [const_iterator](#) __last)
- [iterator](#) **erase** ([const_iterator](#) __it)
- [iterator](#) **erase** ([iterator](#) __it)
- node_type **extract** (const key_type &__key)
- node_type **extract** ([const_iterator](#) __position)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[iterator](#) **find** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[const_iterator](#) **find** (const _Kt &__k) const
- [iterator](#) **find** (const key_type &__key)
- [const_iterator](#) **find** (const key_type &__key) const
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)

- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&::value>::type>
std::pair< iterator, bool > insert (_Pair &&__obj)`
- `std::pair< iterator, bool > insert (const value_type &__obj)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&::value>::type>
iterator insert (const_iterator __hint, _Pair &&__obj)`
- `iterator insert (const_iterator __hint, const value_type &__obj)`
- `iterator insert (const_iterator __hint, node_type &&__nh)`
- `iterator insert (const_iterator __hint, value_type &&__x)`
- `insert_return_type insert (node_type &&__nh)`
- `void insert (std::initializer_list< value_type > __l)`
- `std::pair< iterator, bool > insert (value_type &&__x)`
- `template<typename _Obj >
pair< iterator, bool > insert_or_assign (const key_type &__k, _Obj &&__obj)`
- `template<typename _Obj >
iterator insert_or_assign (const_iterator __hint, const key_type &__k, _Obj &&__obj)`
- `template<typename _Obj >
iterator insert_or_assign (const_iterator __hint, key_type &&__k, _Obj &&__obj)`
- `template<typename _Obj >
pair< iterator, bool > insert_or_assign (key_type &&__k, _Obj &&__obj)`
- `float max_load_factor () const noexcept`
- `void max_load_factor (float __f)`
- `template<typename _H2, typename _P2 >
void merge (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &&__source)`
- `template<typename _H2, typename _P2 >
void merge (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `template<typename _H2, typename _P2 >
void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &&__source)`
- `template<typename _H2, typename _P2 >
void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `unordered_map & operator= (const unordered_map &)=default`
- `unordered_map & operator= (initializer_list< value_type > __l)`
- `unordered_map & operator= (unordered_map &&)=default`
- `void swap (unordered_map &__x) noexcept(noexcept(declval<_Base &>().swap(__x)))`
- `template<typename... _Args>
pair< iterator, bool > try_emplace (const key_type &__k, _Args &&... __args)`
- `template<typename... _Args>
iterator try_emplace (const_iterator __hint, const key_type &__k, _Args &&... __args)`
- `template<typename... _Args>
iterator try_emplace (const_iterator __hint, key_type &&__k, _Args &&... __args)`
- `template<typename... _Args>
pair< iterator, bool > try_emplace (key_type &&__k, _Args &&... __args)`

Protected Member Functions

- `void _M_swap (_Safe_container &__x) noexcept`

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >
class __gnu_debug:: Safe_iterator`
- `template<typename _ItT, typename _SeqT >
class __gnu_debug:: Safe_local_iterator`

6.1041.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal<_to<_Key>,
typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>>>
class std::__debug::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
```

Class std::unordered_map with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [debug/unordered_map](#)

6.1042 std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

```
#include <unordered_map.h>
```

Public Types

- using **insert_return_type** = typename _Hashtable::insert_return_type
- using **node_type** = typename _Hashtable::node_type

- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::mapped_type [mapped_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)

- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- [unordered_map](#) ()=default
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [allocator_type](#) &__a)
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- template<typename _InputIterator >
[unordered_map](#) (_InputIterator __first, _InputIterator __last, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_map](#) (const [allocator_type](#) &__a)
- [unordered_map](#) (const [unordered_map](#) &)=default
- **unordered_map** (const [unordered_map](#) &__umap, const [allocator_type](#) &__a)

- **unordered_map** (initializer_list< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_map** (initializer_list< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** (initializer_list< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (size_type __n, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (unordered_map &&)=default
- **unordered_map** (unordered_map &&__umap, const allocator_type &__a) noexcept(noexcept(_Hashtable(std::move(__umap._M_h), __a)))
- **iterator begin** () noexcept
- **local_iterator begin** (size_type __n)
- **size_type bucket** (const key_type &__key) const
- **size_type bucket_count** () const noexcept
- **size_type bucket_size** (size_type __n) const
- **void clear** () noexcept
- template<typename... _Args>
std::pair< iterator, bool > **emplace** (_Args &&... __args)
- template<typename... _Args>
iterator **emplace_hint** (const_iterator __pos, _Args &&... __args)
- **bool empty** () const noexcept
- **iterator end** () noexcept
- **local_iterator end** (size_type __n)
- **size_type erase** (const key_type &__x)
- **iterator erase** (const_iterator __first, const_iterator __last)
- **node_type extract** (const key_type &__key)
- **node_type extract** (const_iterator __pos)
- **allocator_type get_allocator** () const noexcept
- **hasher hash_function** () const
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- **iterator insert** (const_iterator, node_type &&__nh)
- **void insert** (initializer_list< value_type > __l)
- **insert_return_type insert** (node_type &&__nh)
- template<typename _Obj >
pair< iterator, bool > **insert_or_assign** (const key_type &__k, _Obj &&__obj)
- template<typename _Obj >
iterator **insert_or_assign** (const_iterator __hint, const key_type &__k, _Obj &&__obj)
- template<typename _Obj >
iterator **insert_or_assign** (const_iterator __hint, key_type &&__k, _Obj &&__obj)
- template<typename _Obj >
pair< iterator, bool > **insert_or_assign** (key_type &&__k, _Obj &&__obj)
- **key_equal key_eq** () const
- **float load_factor** () const noexcept
- **size_type max_bucket_count** () const noexcept
- **float max_load_factor** () const noexcept
- **void max_load_factor** (float __z)
- **size_type max_size** () const noexcept
- template<typename _H2, typename _P2 >
void **merge** (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &&__source)

- `template<typename _H2, typename _P2 >`
`void merge (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &&__source)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `unordered_map & operator= (const unordered_map &)=default`
- `unordered_map & operator= (initializer_list< value_type > __l)`
- `unordered_map & operator= (unordered_map &&)=default`
- `void rehash (size_type __n)`
- `void reserve (size_type __n)`
- `size_type size () const noexcept`
- `void swap (unordered_map &__x) noexcept(noexcept(_M_h.swap(__x._M_h)))`
- `template<typename... _Args>`
`pair< iterator, bool > try_emplace (const key_type &__k, _Args &&... __args)`
- `template<typename... _Args>`
`iterator try_emplace (const_iterator __hint, const key_type &__k, _Args &&... __args)`
- `template<typename... _Args>`
`iterator try_emplace (const_iterator __hint, key_type &&__k, _Args &&... __args)`
- `template<typename... _Args>`
`pair< iterator, bool > try_emplace (key_type &&__k, _Args &&... __args)`
- `const_iterator begin () const noexcept`
- `const_iterator cbegin () const noexcept`
- `const_iterator end () const noexcept`
- `const_iterator cend () const noexcept`
- `std::pair< iterator, bool > insert (const value_type &__x)`
- `std::pair< iterator, bool > insert (value_type &&__x)`
- `template<typename _Pair >`
`__enable_if_t< is_constructible< value_type, _Pair && >::value, pair< iterator, bool > > insert (_Pair &&__x)`
- `iterator insert (const_iterator __hint, const value_type &__x)`
- `iterator insert (const_iterator __hint, value_type &&__x)`
- `template<typename _Pair >`
`__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > insert (const_iterator __hint, _Pair &&__x)`
- `iterator erase (const_iterator __position)`
- `iterator erase (iterator __position)`
- `iterator find (const key_type &__x)`

- template<typename _Kt >
auto **find** (const _Kt &__x) -> decltype(_M_h._M_find_tr(__x))
- **const_iterator find** (const **key_type** &__x) const
- template<typename _Kt >
auto **find** (const _Kt &__x) const -> decltype(_M_h._M_find_tr(__x))
- **size_type count** (const **key_type** &__x) const
- template<typename _Kt >
auto **count** (const _Kt &__x) const -> decltype(_M_h._M_count_tr(__x))
- **bool contains** (const **key_type** &__x) const
- template<typename _Kt >
auto **contains** (const _Kt &__x) const -> decltype(_M_h._M_find_tr(__x), void(), true)
- **std::pair< iterator, iterator > equal_range** (const **key_type** &__x)
- template<typename _Kt >
auto **equal_range** (const _Kt &__x) -> decltype(_M_h._M_equal_range_tr(__x))
- **std::pair< const_iterator, const_iterator > equal_range** (const **key_type** &__x) const
- template<typename _Kt >
auto **equal_range** (const _Kt &__x) const -> decltype(_M_h._M_equal_range_tr(__x))
- **mapped_type & operator[]** (const **key_type** &__k)
- **mapped_type & operator[]** (**key_type** &&__k)
- **mapped_type & at** (const **key_type** &__k)
- **const mapped_type & at** (const **key_type** &__k) const
- **const_local_iterator begin** (**size_type** __n) const
- **const_local_iterator cbegin** (**size_type** __n) const
- **const_local_iterator end** (**size_type** __n) const
- **const_local_iterator cend** (**size_type** __n) const

Friends

- template<typename _Key1, typename _Tp1, typename _Hash1, typename _Pred1, typename _Alloc1 >
bool **operator==** (const **unordered_map**<_Key1, _Tp1, _Hash1, _Pred1, _Alloc1> &, const **unordered_map**<_Key1, _Tp1, _Hash1, _Pred1, _Alloc1> &)
- template<typename, typename, typename >
class **std::_Hash_merge_helper**

6.1042.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_↵  
_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
class std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
```

A standard container composed of unique keys (containing at most one of each key value) that associates values of another type with the keys.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Hash</code>	Hashing function object type, defaults to <code>hash<_Value></code> .
<code>_Pred</code>	Predicate function object type, defaults to <code>equal_to<_Value></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>std::allocator<std::pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), and [unordered associative container](#)

The resulting value type of the container is `std::pair<const _Key, _Tp>`.

Base is `_Hashtable`, dispatched at compile time via template alias `__umap_hashtable`.

6.1042.2 Member Typedef Documentation

allocator_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::allocator_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::allocator_↵
_type
```

Public typedefs.

const_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::const_↵
_iterator
```

Iterator-related typedefs.

const_local_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >_↵
::const_local_iterator
```

Iterator-related typedefs.

const_pointer

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_pointer std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::const_↵
_pointer
```

Iterator-related typedefs.

const_reference

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_reference std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::const_↵
_reference
```

Iterator-related typedefs.

difference_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::difference_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::difference_↵
_type
Iterator-related typedefs.
```

hasher

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::hasher std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::hasher
Public typedefs.
```

iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::iterator
Iterator-related typedefs.
```

key_equal

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::key_equal std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::key_equal
Public typedefs.
```

key_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::key_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::key_type
Public typedefs.
```

local_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::local_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::local_↵
_iterator
Iterator-related typedefs.
```

mapped_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::mapped_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::mapped_↵
type
Public typedefs.
```

pointer

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::pointer std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::pointer
Iterator-related typedefs.
```

reference

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::reference std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::reference
Iterator-related typedefs.
```

size_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::size_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::size_type
Iterator-related typedefs.
```

value_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::value_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::value_type
Public typedefs.
```

6.1042.3 Constructor & Destructor Documentation**unordered_map() [1/7]**

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map ( ) [default]
Default constructor.
```

unordered_map() [2/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    size_type __n,
    const hasher & __hf = hasher(),
    const key_equal & __eqf = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

unordered_map() [3/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _InputIterator >
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    _InputIterator __first,
    _InputIterator __last,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_map consisting of copies of the elements from [`__first`,`__last`). This is linear in N (where N is distance(`__first`,`__last`)).

unordered_map() [4/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > & ) [default]
```

Copy constructor.

unordered_map() [5/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > && ) [default]
```

Move constructor.

unordered_map() [6/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    const allocator_type & __a ) [inline], [explicit]
```

Creates an unordered_map with no elements.

Parameters

<code>↵</code>	An allocator object.
<code>__a</code>	

unordered_map() [7/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_map (
    initializer_list< value_type > __l,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_map from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_map consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

6.1042.4 Member Function Documentation**at()** [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
mapped_type & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::at (
    const key_type & __k ) [inline]
```

Access to unordered_map data.

Parameters

<code>↵ __k</code>	The key for which data should be retrieved.
------------------------	---

Returns

A reference to the data whose key is equal to `__k`, if such a data is present in the unordered_map.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

at() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const mapped_type & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::at (
    const key_type & __k ) const [inline]
```

Access to unordered_map data.

Parameters

<code>↔ _k</code>	The key for which data should be retrieved.
-----------------------	---

Returns

A reference to the data whose key is equal to `__k`, if such a data is present in the `unordered_map`.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

begin() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::begin ( ) const [inline],  
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_map`.

begin() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the `unordered_map`.

begin() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::begin (   
    size_type __n ) [inline]
```

Returns a read/write iterator pointing to the first bucket element.

Parameters

<code>↔ _n</code>	The bucket index.
-----------------------	-------------------

Returns

A read/write local iterator.

begin() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
const_local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::begin (   
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>_↔</code>	The bucket index.
<code>_n</code>	

Returns

A read-only local iterator.

bucket_count()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::bucket_count ( ) const [inline],
[noexcept]
```

Returns the number of buckets of the unordered_map.

cbegin() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::cbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_map.

cbegin() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_local_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::cbegin (
size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>_↔</code>	The bucket index.
<code>_n</code>	

Returns

A read-only local iterator.

cend() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::cend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_map.

Referenced by `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert_or_assign()`, and `std::unordered_map< _Key, _Tp, _Hash`

cend() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
```

```
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::cend (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

clear()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in an unordered_map. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x), void(), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__x</code>	Key of elements to be located.
------------------	--------------------------------

Returns

True if there is any element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
bool std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__x</code>	Key of elements to be located.
------------------	--------------------------------

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
template<typename _Kt >  
auto std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::count (  
    const _Kt & __x ) const -> decltype(_M_h._M_count_tr(__x))    [inline]
```

Finds the number of elements.

Parameters

<div><div><div><div><div><div>↵</div></div></div><div><div><div>_</div><div>x</div></div></div></div></div></div>	Key to count.
--	---------------

Returns

Number of elements with specified key.

This function only makes sense for unordered_multimap; for unordered_map the result will either be 0 (not present) or 1 (present).

count() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
size_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::count (  
    const key_type & __x ) const    [inline]
```

Finds the number of elements.

Parameters

<div><div><div><div><div><div>↵</div></div></div><div><div><div>_</div><div>x</div></div></div></div></div></div>	Key to count.
--	---------------

Returns

Number of elements with specified key.

This function only makes sense for unordered_multimap; for unordered_map the result will either be 0 (not present) or 1 (present).

emplace()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
template<typename... _Args>  
std::pair< iterator, bool > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::emplace (  
    _Args &&... __args )    [inline]
```

Attempts to build and insert a std::pair into the unordered_map.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the `unordered_map`. An `unordered_map` relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the `unordered_map`. Insertion requires amortized constant time.

emplace_hint()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
template<typename... _Args>
iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the `unordered_map`.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the `std::pair` built from `__args` (may or may not be that `std::pair`).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

empty()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
bool std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::empty ( ) const [inline], [noexcept]
Returns true if the unordered_map is empty.
```

end() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::end ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_map`.

end() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::end ( ) [inline], [noexcept]  
Returns a read/write iterator that points one past the last element in the unordered_map.  
Referenced by std::unordered\_map< \_Key, \_Tp, \_Hash, \_Pred, \_Alloc >::extract\(\).
```

end() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
local_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::end (   
    size_type __n ) [inline]  
Returns a read/write iterator pointing to one past the last bucket elements.
```

Parameters

_↵	The bucket index.
_n	

Returns

A read/write local iterator.

end() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
const_local_iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::end (   
    size_type __n ) const [inline]  
Returns a read-only (constant) iterator pointing to one past the last bucket elements.
```

Parameters

_↵	The bucket index.
_n	

Returns

A read-only local iterator.

equal_range() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
template<typename _Kt >  
auto std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (   
    const _Kt & __x ) -> decltype(_M_h._M_equal_range_tr(__x)) [inline]  
Finds a subsequence matching given key.
```

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for unordered_multimap.

equal_range() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(_M_h._M_equal_range_tr(__x))    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for unordered_multimap.

equal_range() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::pair< iterator, iterator > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::equal_↔
range (
    const key_type & __x )    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for unordered_multimap.

equal_range() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
```



```
std::pair< const_iterator, const_iterator > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc
>::equal_range (
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for `unordered_multimap`.

erase() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an `unordered_map`. For an `unordered_map` the result of this function can only be 0 (not present) or 1 (present). Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [`__first`,`__last`) range of elements from an `unordered_map`.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an `unordered_map`. Note that this function only erases the elements, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from an `unordered_map`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_map`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from an `unordered_map`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_map`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
node_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::extract (
    const key_type & __key ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
node_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::end()`.

find() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) -> decltype(_M_h._M_find_tr(__x)) [inline]
```

Tries to locate an element in an `unordered_map`.

Parameters

<code>_↵</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x)) [inline]
```

Tries to locate an element in an `unordered_map`.

Parameters

<code>_↵</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
```

```
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) [inline]
```

Tries to locate an element in an unordered_map.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in an unordered_map.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

get_allocator()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
allocator_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::get_allocator ( ) const
[inline], [noexcept]
```

Returns the allocator object used by the unordered_map.

hash_function()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
hasher std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::hash_function ( ) const [inline]
```

Returns the hash functor object with which the unordered_map was constructed.

insert() [1/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
```

```
template<typename _InputIterator >
void std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair && >::value, pair< iterator, bool > > std::unordered_map<
_Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    _Pair && __x ) [inline]
```

Attempts to insert a `std::pair` into the `unordered_map`.

Parameters

<code>↵</code> <code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
------------------------------------	---

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the `unordered_map`. An `unordered_map` relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the `unordered_map`. Insertion requires amortized constant time.

insert() [3/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::pair< iterator, bool > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Attempts to insert a `std::pair` into the `unordered_map`.

Parameters

<code>↵</code> <code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
------------------------------------	---

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map. Insertion requires amortized constant time.

insert() [4/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > std::unordered_map<
_Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    _Pair && __x ) [inline]
```

Attempts to insert a std::pair into the unordered_map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument insert() does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

insert() [5/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    const value_type & __x ) [inline]
```

Attempts to insert a std::pair into the unordered_map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument insert() does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

insert() [6/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    value_type && __x ) [inline]
```

Attempts to insert a `std::pair` into the `unordered_map`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

References [std::move\(\)](#).

insert() [7/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator ,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [8/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of elements into the `unordered_map`.

Parameters

↵ _↵ ↵ _↵ /	A <code>std::initializer_list<value_type></code> of elements to be inserted.
-------------------------	--

Complexity similar to that of the range constructor.

insert() [9/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
insert_return_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [10/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::pair< iterator, bool > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    value_type && __x ) [inline]
```

Attempts to insert a std::pair into the unordered_map.

Parameters

_↵	Pair to be inserted (see std::make_pair for easy creation of pairs).
_X	

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map.

Insertion requires amortized constant time.

References [std::move\(\)](#).

insert_or_assign() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Obj >
pair< iterator, bool > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::insert_or_assign (
    const key_type & __k,
    _Obj && __obj ) [inline]
```

Attempts to insert a std::pair into the unordered_map.

Parameters

__k	Key to use for finding a possibly existing pair in the map.
__obj	Argument used to generate the .second for a pair instance.

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map. If the pair was already in the unordered_map, the .second of the pair is assigned from __obj.

Insertion requires amortized constant time.

References `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::cend()`.

insert_or_assign() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Obj >
iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::insert_or_assign (
    const_iterator __hint,
    const key_type & __k,
    _Obj && __obj ) [inline]
```

Attempts to insert a `std::pair` into the `unordered_map`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__k</code>	Key to use for finding a possibly existing pair in the <code>unordered_map</code> .
<code>__obj</code>	Argument used to generate the .second for a pair instance.

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does.

If the pair was already in the unordered map, the .second of the pair is assigned from `__obj`. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

key_eq()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
key_equal std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::key_eq ( ) const [inline]
```

Returns the key comparison object with which the `unordered_map` was constructed.

load_factor()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
float std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::load_factor ( ) const [inline],
[noexcept]
```

Returns the average number of elements per bucket.

max_bucket_count()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::max_bucket_count ( ) const [inline],
[noexcept]
```

Returns the maximum number of buckets of the `unordered_map`.

max_load_factor() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
float std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::max_load_factor ( ) const [inline],
[noexcept]
```

Returns a positive number that the unordered_map tries to keep the load factor less than or equal to.

max_load_factor() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::max_load_factor (
    float __z ) [inline]
```

Change the unordered_map maximum load factor.

Parameters

↵	The new maximum load factor.
__z	

max_size()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::max_size ( ) const [inline],
[noexcept]
```

Returns the maximum size of the unordered_map.

operator=() [1/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
unordered_map & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > & ) [default]
```

Copy assignment operator.

operator=() [2/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
unordered_map & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Unordered_map list assignment operator.

Parameters

↵	An initializer_list.
↵	
↵	
↵	
/	

This function fills an unordered_map with copies of the elements in the initializer list __l.

Note that the assignment completely changes the `unordered_map` and that the resulting `unordered_map`'s size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
unordered_map & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > && ) [default]
```

Move assignment operator.

operator[]() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
mapped_type & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::operator[] (
    const key_type & __k ) [inline]
```

Subscript (`[]`) access to `unordered_map` data.

Parameters

<code>_↵</code> <code>_k</code>	The key for which data should be retrieved.
------------------------------------	---

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript (`[]`) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned. Lookup requires constant time.

operator[]() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
mapped_type & std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::operator[] (
    key_type && __k ) [inline]
```

Subscript (`[]`) access to `unordered_map` data.

Parameters

<code>_↵</code> <code>_k</code>	The key for which data should be retrieved.
------------------------------------	---

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript (`[]`) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned. Lookup requires constant time. References `std::move()`.

rehash()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::rehash (
    size_type __n ) [inline]
```

May rehash the unordered_map.

Parameters

_↔ _n	The new number of buckets.
------------------	----------------------------

Rehash will occur only if the new number of buckets respect the unordered_map maximum load factor.

reserve()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::reserve (
    size_type __n ) [inline]
```

Prepare the unordered_map for a specified number of elements.

Parameters

_↔ _n	Number of elements required.
------------------	------------------------------

Same as rehash(ceil(n / max_load_factor())).

size()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
size_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the size of the unordered_map.

swap()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::swap (
    unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another unordered_map.

Parameters

_↔ _x	An unordered_map of the same element and allocator types.
------------------	---

This exchanges the elements between two unordered_map in constant time. Note that the global std::swap() function is specialized such that std::swap(m1,m2) will feed to this function.

try_emplace() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename... _Args>
pair< iterator, bool > std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::try_emplace (
    const key_type & __k,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_map.

Parameters

<code>__k</code>	Key to use for finding a possibly existing pair in the unordered_map.
<code>__args</code>	Arguments used to generate the .second for a new pair instance.

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map. If a pair is not inserted, this function has no effect.

Insertion requires amortized constant time.

References [std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::cend\(\)](#).

try_emplace() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename... _Args>
iterator std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >::try_emplace (
    const_iterator __hint,
    const key_type & __k,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__k</code>	Key to use for finding a possibly existing pair in the unordered_map.
<code>__args</code>	Arguments used to generate the .second for a new pair instance.

Returns

An iterator that points to the element with key of the std::pair built from `__args` (may or may not be that std::pair).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. However, if insertion did not take place, this function has no effect. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

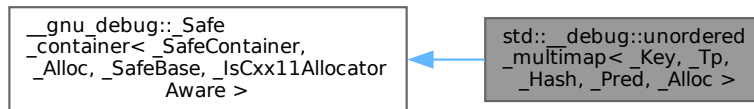
Insertion requires amortized constant time.

The documentation for this class was generated from the following file:

- [unordered_map.h](#)

6.1043 std::__debug::unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc> Class Template Reference

Inheritance diagram for std::__debug::unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>:



Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef __gnu_debug::__Safe_iterator<_Base_const_iterator, unordered_multimap> **const_iterator**
- typedef __gnu_debug::__Safe_local_iterator<_Base_const_local_iterator, unordered_multimap> **const_local_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef _Base::difference_type **difference_type**
- typedef _Base::hasher **hasher**
- typedef __gnu_debug::__Safe_iterator<_Base_iterator, unordered_multimap> **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef __gnu_debug::__Safe_local_iterator<_Base_local_iterator, unordered_multimap> **local_iterator**
- typedef _Base::mapped_type **mapped_type**
- using **node_type** = typename _Base::node_type
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_multimap** (_Base_ref __x)
- template<typename _InputIterator>
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator>
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator>
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (const allocator_type &__a)
- **unordered_multimap** (const unordered_multimap &)=default
- **unordered_multimap** (const unordered_multimap &__umap, const allocator_type &__a)

- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (size_type __n, const allocator_type &__a)
- **unordered_multimap** (size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_multimap** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** ([unordered_multimap](#) &&)=default
- **unordered_multimap** ([unordered_multimap](#) &&__umap, const allocator_type &__a) noexcept(noexcept([_Base](#)(std::move([__umap](#)), [__a](#))))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) **begin** () const noexcept
- [iterator](#) **begin** () noexcept
- [local_iterator](#) **begin** (size_type __b)
- [const_local_iterator](#) **begin** (size_type __b) const
- size_type **bucket_size** (size_type __b) const
- [const_iterator](#) **cbegin** () const noexcept
- [const_local_iterator](#) **cbegin** (size_type __b) const
- [const_iterator](#) **cend** () const noexcept
- [const_local_iterator](#) **cend** (size_type __b) const
- void **clear** () noexcept
- template<typename... _Args>
[iterator](#) **emplace** (_Args &&... __args)
- template<typename... _Args>
[iterator](#) **emplace_hint** (const [iterator](#) __hint, _Args &&... __args)
- [const_iterator](#) **end** () const noexcept
- [iterator](#) **end** () noexcept
- [local_iterator](#) **end** (size_type __b)
- [const_local_iterator](#) **end** (size_type __b) const
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[std::pair](#)< const [iterator](#), const [iterator](#) > **equal_range** (const _Kt &__k) const
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const key_type &__key)
- [std::pair](#)< const [iterator](#), const [iterator](#) > **equal_range** (const key_type &__key) const
- [_Base_iterator](#) **erase** ([_Base_const_iterator](#) __it)
- size_type **erase** (const key_type &__key)
- [iterator](#) **erase** (const [iterator](#) __first, const [iterator](#) __last)
- [iterator](#) **erase** (const [iterator](#) __it)
- [iterator](#) **erase** ([iterator](#) __it)
- node_type **extract** (const key_type &__key)
- node_type **extract** (const [iterator](#) __position)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
[iterator](#) **find** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
const [iterator](#) **find** (const _Kt &__k) const
- [iterator](#) **find** (const key_type &__key)
- [const_iterator](#) **find** (const key_type &__key) const

- `template<typename _InputIterator >`
`void insert (_InputIterator __first, _InputIterator __last)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`iterator insert (_Pair &&__obj)`
- `iterator insert (const value_type &__obj)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`iterator insert (const_iterator __hint, _Pair &&__obj)`
- `iterator insert (const_iterator __hint, const value_type &__obj)`
- `iterator insert (const_iterator __hint, node_type &&__nh)`
- `iterator insert (const_iterator __hint, value_type &&__x)`
- `iterator insert (node_type &&__nh)`
- `void insert (std::initializer_list< value_type > __l)`
- `iterator insert (value_type &&__x)`
- `float max_load_factor ()` `const noexcept`
- `void max_load_factor (float __f)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &&__source)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &&__source)`
- `template<typename _H2, typename _P2 >`
`void merge (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &__source)`
- `unordered_multimap & operator= (const unordered_multimap &)=default`
- `unordered_multimap & operator= (initializer_list< value_type > __l)`
- `unordered_multimap & operator= (unordered_multimap &&)=default`
- `void swap (unordered_multimap &__x) noexcept(noexcept(declval< _Base & >().swap(__x)))`

Protected Member Functions

- `void _M_swap (_Safe_container &__x) noexcept`

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >`
`class ::__gnu_debug:: Safe_iterator`
- `template<typename _ItT, typename _SeqT >`
`class ::__gnu_debug:: Safe_local_iterator`

6.1043.1 Detailed Description

`template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal<_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>>>`
`class std::__debug::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`

Class std::unordered_multimap with safety/checking/debug instrumentation.
The documentation for this class was generated from the following file:

- `debug/unordered_map`

6.1044 std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

```
#include <unordered_map.h>
```


Public Types

- using **node_type** = typename _Hashtable::node_type
- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::mapped_type [mapped_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)
- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- [unordered_multimap](#) ()=default
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [allocator_type](#) &__a)
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- template<typename _InputIterator >
[unordered_multimap](#) (_InputIterator __first, _InputIterator __last, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_multimap](#) (const [allocator_type](#) &__a)
- [unordered_multimap](#) (const [unordered_multimap](#) &)=default
- **unordered_multimap** (const [unordered_multimap](#) &__ummap, const [allocator_type](#) &__a)
- **unordered_multimap** (initializer_list< [value_type](#) > __l, [size_type](#) __n, const [allocator_type](#) &__a)
- **unordered_multimap** (initializer_list< [value_type](#) > __l, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_multimap](#) (initializer_list< [value_type](#) > __l, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- **unordered_multimap** ([size_type](#) __n, const [allocator_type](#) &__a)
- **unordered_multimap** ([size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_multimap](#) ([size_type](#) __n, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_multimap](#) ([unordered_multimap](#) &&)=default
- **unordered_multimap** ([unordered_multimap](#) &&__ummap, const [allocator_type](#) &__a) noexcept(noexcept(_↵
_Hashtable(std::move(__ummap._M_h), __a)))
- [iterator](#) [begin](#) () noexcept
- [local_iterator](#) [begin](#) ([size_type](#) __n)

- `size_type bucket` (const `key_type` &__key) const
 - `size_type bucket_count` () const noexcept
 - `size_type bucket_size` (size_type __n) const
 - `void clear` () noexcept
 - `template<typename... _Args>`
`iterator emplace` (_Args &&... __args)
 - `template<typename... _Args>`
`iterator emplace_hint` (const_iterator __pos, _Args &&... __args)
 - `bool empty` () const noexcept
 - `iterator end` () noexcept
 - `local_iterator end` (size_type __n)
 - `size_type erase` (const `key_type` &__x)
 - `iterator erase` (const_iterator __first, const_iterator __last)
 - `node_type extract` (const `key_type` &__key)
 - `node_type extract` (const_iterator __pos)
 - `allocator_type get_allocator` () const noexcept
 - `hasher hash_function` () const
 - `template<typename _InputIterator >`
`void insert` (_InputIterator __first, _InputIterator __last)
 - `iterator insert` (const_iterator __hint, node_type &&__nh)
 - `void insert` (initializer_list< `value_type` > __l)
 - `iterator insert` (node_type &&__nh)
 - `key_equal key_eq` () const
 - `float load_factor` () const noexcept
 - `size_type max_bucket_count` () const noexcept
 - `float max_load_factor` () const noexcept
 - `void max_load_factor` (float __z)
 - `size_type max_size` () const noexcept
 - `template<typename _H2, typename _P2 >`
`void merge` (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &&__source)
 - `template<typename _H2, typename _P2 >`
`void merge` (unordered_map< _Key, _Tp, _H2, _P2, _Alloc > &__source)
 - `template<typename _H2, typename _P2 >`
`void merge` (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &&__source)
 - `template<typename _H2, typename _P2 >`
`void merge` (unordered_multimap< _Key, _Tp, _H2, _P2, _Alloc > &__source)
 - `unordered_multimap & operator=` (const unordered_multimap &)=default
 - `unordered_multimap & operator=` (initializer_list< `value_type` > __l)
 - `unordered_multimap & operator=` (unordered_multimap &&)=default
 - `void rehash` (size_type __n)
 - `void reserve` (size_type __n)
 - `size_type size` () const noexcept
 - `void swap` (unordered_multimap &__x) noexcept(noexcept(_M_h.swap(__x._M_h)))
-
- `const_iterator begin` () const noexcept
 - `const_iterator cbegin` () const noexcept
-
- `const_iterator end` () const noexcept
 - `const_iterator cend` () const noexcept

- `iterator insert` (`const value_type &__x`)
- `iterator insert` (`value_type &&__x`)
- `template<typename _Pair >`
`__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > insert` (`_Pair &&__x`)
- `iterator insert` (`const_iterator __hint, const value_type &__x`)
- `iterator insert` (`const_iterator __hint, value_type &&__x`)
- `template<typename _Pair >`
`__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > insert` (`const_iterator __hint, _Pair &&__x`)
- `iterator erase` (`const_iterator __position`)
- `iterator erase` (`iterator __position`)
- `iterator find` (`const key_type &__x`)
- `template<typename _Kt >`
`auto find` (`const _Kt &__x`) -> `decltype(_M_h._M_find_tr(__x))`
- `const_iterator find` (`const key_type &__x`) `const`
- `template<typename _Kt >`
`auto find` (`const _Kt &__x`) `const` -> `decltype(_M_h._M_find_tr(__x))`
- `size_type count` (`const key_type &__x`) `const`
- `template<typename _Kt >`
`auto count` (`const _Kt &__x`) `const` -> `decltype(_M_h._M_count_tr(__x))`
- `bool contains` (`const key_type &__x`) `const`
- `template<typename _Kt >`
`auto contains` (`const _Kt &__x`) `const` -> `decltype(_M_h._M_find_tr(__x), void(), true)`
- `std::pair< iterator, iterator > equal_range` (`const key_type &__x`)
- `template<typename _Kt >`
`auto equal_range` (`const _Kt &__x`) -> `decltype(_M_h._M_equal_range_tr(__x))`
- `std::pair< const_iterator, const_iterator > equal_range` (`const key_type &__x`) `const`
- `template<typename _Kt >`
`auto equal_range` (`const _Kt &__x`) `const` -> `decltype(_M_h._M_equal_range_tr(__x))`
- `const_local_iterator begin` (`size_type __n`) `const`
- `const_local_iterator cbegin` (`size_type __n`) `const`
- `const_local_iterator end` (`size_type __n`) `const`
- `const_local_iterator cend` (`size_type __n`) `const`

Friends

- template<typename _Key1, typename _Tp1, typename _Hash1, typename _Pred1, typename _Alloc1 >
bool **operator==** (const unordered_multimap<_Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &, const unordered_multimap<_Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &)
- template<typename , typename , typename >
class std::_Hash_merge_helper

6.1044.1 Detailed Description

template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
class std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >

A standard container composed of equivalent keys (possibly containing multiple of each key value) that associates values of another type with the keys.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Hash</code>	Hashing function object type, defaults to hash<_Value>.
<code>_Pred</code>	Predicate function object type, defaults to equal_to<_Value>.
<code>_Alloc</code>	Allocator type, defaults to std::allocator<std::pair<const _Key, _Tp>>.

Meets the requirements of a [container](#), and [unordered associative container](#)

The resulting value type of the container is std::pair<const _Key, _Tp>.

Base is _Hashtable, dispatched at compile time via template alias __ummap_hashtable.

6.1044.2 Member Typedef Documentation

allocator_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>,
typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::allocator_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::allocator_type
```

Public typedefs.

const_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>,
typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::const_iterator
```

Iterator-related typedefs.

const_local_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>,
typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
typedef _Hashtable::const_local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::const_local_iterator
```

Iterator-related typedefs.

const_pointer

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::const_pointer std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵  
::const_pointer
```

Iterator-related typedefs.

const_reference

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::const_reference std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵  
::const_reference
```

Iterator-related typedefs.

difference_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::difference_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵  
::difference_type
```

Iterator-related typedefs.

hasher

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::hasher std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::hasher
```

Public typedefs.

iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::iterator
```

Iterator-related typedefs.

key_equal

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::key_equal std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::key_↵  
equal
```

Public typedefs.

key_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
typedef _Hashtable::key_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::key_type
```

Public typedefs.

local_iterator

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵
::local_iterator
```

Iterator-related typedefs.

mapped_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::mapped_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵
::mapped_type
```

Public typedefs.

pointer

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::pointer std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::pointer
```

Iterator-related typedefs.

reference

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::reference std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::reference
```

Iterator-related typedefs.

size_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::size_↵
type
```

Iterator-related typedefs.

value_type

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
typedef _Hashtable::value_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::value_↵
_type
```

Public typedefs.

6.1044.3 Constructor & Destructor Documentation**unordered_multimap()** [1/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap ( ) [default]
```

Default constructor.

unordered_multimap() [2/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    size_type __n,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

unordered_multimap() [3/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _InputIterator >
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    _InputIterator __first,
    _InputIterator __last,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_multimap from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multimap consisting of copies of the elements from [`__first`,`__last`). This is linear in N (where N is `distance(__first,__last)`).

unordered_multimap() [4/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > & ) [default]
```

Copy constructor.

unordered_multimap() [5/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > && ) [default]
```

Move constructor.

unordered_multimap() [6/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    const allocator_type & __a ) [inline], [explicit]
```

Creates an unordered_multimap with no elements.

Parameters

<code>_↵</code>	An allocator object.
<code>_a</code>	

unordered_multimap() [7/7]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::unordered_multimap (
    initializer_list< value_type > __l,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_multimap from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multimap consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

6.1044.4 Member Function Documentation**begin()** [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::begin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multimap.

begin() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the unordered_multimap.

begin() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::begin (   
    size_type __n ) [inline]
```

Returns a read/write iterator pointing to the first bucket element.

Parameters

<code>_↵</code>	The bucket index.
<code>_n</code>	

Returns

A read/write local iterator.

begin() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
const_local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::begin (   
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>_↵</code>	The bucket index.
<code>_n</code>	

Returns

A read-only local iterator.

bucket_count()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>  
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::bucket_count ( ) const  
[inline], [noexcept]
```

Returns the number of buckets of the unordered_multimap.

cbegin() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
```

```
const_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::cbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multimap.

cbegin() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::cbegin (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_↵$	The bucket index.
$_n$	

Returns

A read-only local iterator.

cend() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::cend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_multimap.

Referenced by [std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert\(\)](#).

cend() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::cend (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_↵$	The bucket index.
$_n$	

Returns

A read-only local iterator.

clear()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in an `unordered_multimap`. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x), void(), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_↵</code> <code>_X</code>	Key of elements to be located.
------------------------------------	--------------------------------

Returns

True if there is any element with the specified key.

contains() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
bool std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>_↵</code> <code>_X</code>	Key of elements to be located.
------------------------------------	--------------------------------

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::count (
    const _Kt & __x ) const -> decltype(_M_h._M_count_tr(__x)) [inline]
```

Finds the number of elements.

Parameters

<code>_↵</code> <code>_X</code>	Key to count.
------------------------------------	---------------

Returns

Number of elements with specified key.

count() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::count (
    const key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

↵ _x	Key to count.
-----------------------	---------------

Returns

Number of elements with specified key.

emplace()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename... _Args>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::emplace (
    _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_multimap.

Parameters

__args	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).
---------------	---

Returns

An iterator that points to the inserted pair.

This function attempts to build and insert a (key, value) pair into the unordered_multimap. Insertion requires amortized constant time.

emplace_hint()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename... _Args>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_multimap.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the `std::pair` built from `__args`.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

empty()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
bool std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the `unordered_multimap` is empty.

end() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::end ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_multimap`.

end() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::end ( ) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the `unordered_multimap`.

Referenced by `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::extract()`.

end() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
local_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::end (
    size_type __n ) [inline]
```

Returns a read/write iterator pointing to one past the last bucket elements.

Parameters

<code>__↵</code>	The bucket index.
<code>__n</code>	

Returns

A read/write local iterator.

end() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
const_local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::end (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

↵ _n	The bucket index.
-----------------------	-------------------

Returns

A read-only local iterator.

equal_range() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
template<typename _Kt >
auto std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(_M_h._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

↵ _x	Key to be located.
-----------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
template<typename _Kt >
auto std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(_M_h._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

↵ _x	Key to be located.
-----------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
std::pair< iterator, iterator > std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >↵  
::equal_range (   
    const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

↵ _X	Key to be located.
-------------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
std::pair< const_iterator, const_iterator > std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _↵  
Alloc >::equal_range (   
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

↵ _X	Key to be located.
-------------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

erase() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵  
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>  
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (   
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

↵ _X	Key of elements to be erased.
-------------------------	-------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an unordered_multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [*__first*,*__last*) range of elements from an unordered_multimap.

Parameters

<i>__first</i>	Iterator pointing to the start of the range to be erased.
<i>__last</i>	Iterator pointing to the end of the range to be erased.

Returns

The iterator *__last*.

This function erases a sequence of elements from an unordered_multimap. Note that this function only erases the elements, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from an unordered_multimap.

Parameters

<i>__position</i>	An iterator pointing to the element to be erased.
-------------------	---

Returns

An iterator pointing to the element immediately following *__position* prior to the element being erased. If no such element exists, end() is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
```



```
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from an unordered_multimap.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
node_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::extract (
    const key_type & __key ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
node_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::end()`.

find() [1/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↔
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) -> decltype(_M_h._M_find_tr(__x)) [inline]
```

Tries to locate an element in an unordered_multimap.

Parameters

<code>_↔</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Kt >
auto std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x))    [inline]
```

Tries to locate an element in an unordered_multimap.

Parameters

↵ _X	Key to be located.
----------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [3/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) [inline]
```

Tries to locate an element in an unordered_multimap.

Parameters

↵ _X	Key to be located.
----------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [4/4]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
const_iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in an unordered_multimap.

Parameters

↵ _X	Key to be located.
----------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

get_allocator()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
allocator_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::get_allocator ( )
const [inline], [noexcept]
```

Returns the allocator object used by the unordered_multimap.

hash_function()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
hasher std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::hash_function ( ) const [inline]
```

Returns the hash functor object with which the unordered_multimap was constructed.

insert() [1/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _InputIterator >
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > std::unordered_multimap<
_Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    _Pair && __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__↵</code> <code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
--------------------------------------	--

Returns

An iterator that points to the inserted pair.

Insertion requires amortized constant time.

insert() [3/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
------------------	--

Returns

An iterator that points to the inserted pair.

Insertion requires amortized constant time.

insert() [4/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
template<typename _Pair >
__enable_if_t< is_constructible< value_type, _Pair && >::value, iterator > std::unordered_multimap<
_Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    _Pair && __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↵associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

insert() [5/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
```

```
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    const value_type & __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

insert() [6/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [7/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    value_type && __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

References [std::move\(\)](#).

insert() [8/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (
    initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of elements into the unordered_multimap.

Parameters

↵	A std::initializer_list<value_type> of elements to be inserted.
_↵	
↵	
_↵	
/	

Complexity similar to that of the range constructor.

insert() [9/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>::cend\(\)](#), and [std::move\(\)](#).

insert() [10/10]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (
    value_type && __x ) [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

_↵	Pair to be inserted (see std::make_pair for easy creation of pairs).
_X	

Returns

An iterator that points to the inserted pair.

Insertion requires amortized constant time.

References [std::move\(\)](#).

key_eq()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
key_equal std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::key_eq ( ) const [inline]
```

Returns the key comparison object with which the unordered_multimap was constructed.

load_factor()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
float std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::load_factor ( ) const [inline],
[noexcept]
```

Returns the average number of elements per bucket.

max_bucket_count()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::max_bucket_count ( ) const
[inline], [noexcept]
```

Returns the maximum number of buckets of the unordered_multimap.

max_load_factor() [1/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
float std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::max_load_factor ( ) const [inline],
[noexcept]
```

Returns a positive number that the unordered_multimap tries to keep the load factor less than or equal to.

max_load_factor() [2/2]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::max_load_factor (
    float __z ) [inline]
```

Change the unordered_multimap maximum load factor.

Parameters

<code>_↵</code>	The new maximum load factor.
<code>_z</code>	

max_size()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::max_size ( ) const [inline],
[noexcept]
```

Returns the maximum size of the unordered_multimap.

operator=() [1/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>>
unordered_multimap & std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > & ) [default]
```

Copy assignment operator.

operator=() [2/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
unordered_multimap & std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Unordered_multimap list assignment operator.

Parameters

↵	An initializer_list.
_↵	
↵	
_↵	
/	

This function fills an unordered_multimap with copies of the elements in the initializer list __l.

Note that the assignment completely changes the unordered_multimap and that the resulting unordered_multimap's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
unordered_multimap & std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::operator= (
    unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > && ) [default]
```

Move assignment operator.

rehash()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::rehash (
    size_type __n ) [inline]
```

May rehash the unordered_multimap.

Parameters

_↵	The new number of buckets.
_n	

Rehash will occur only if the new number of buckets respect the unordered_multimap maximum load factor.

reserve()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::reserve (
    size_type __n ) [inline]
```

Prepare the unordered_multimap for a specified number of elements.

Parameters

_↵	Number of elements required.
_n	

Same as `rehash(ceil(n / max_load_factor()))`.

size()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::size ( ) const [inline],
[noexcept]
```

Returns the size of the `unordered_multimap`.

swap()

```
template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_↵
to<_Key>, typename _Alloc = allocator<std::pair<const _Key, _Tp>>>
void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::swap (
    unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another `unordered_multimap`.

Parameters

<code>↵</code> <code>__x</code>	An <code>unordered_multimap</code> of the same element and allocator types.
------------------------------------	---

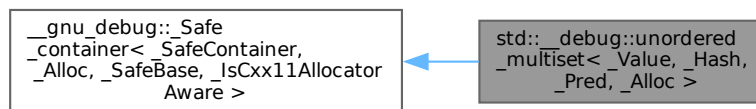
This exchanges the elements between two `unordered_multimap` in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(m1,m2)` will feed to this function.

The documentation for this class was generated from the following file:

- [unordered_map.h](#)

6.1045 std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference

Inheritance diagram for `std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< _Base_const_iterator, unordered_multiset >` **const_iterator**
- typedef `__gnu_debug::Safe_local_iterator< _Base_const_local_iterator, unordered_multiset >` **const_local_↵_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `_Base::difference_type` **difference_type**
- typedef `_Base::hasher` **hasher**

- typedef [__gnu_debug::Safe_iterator](#)< [_Base_iterator](#), [unordered_multiset](#) > [iterator](#)
- typedef [_Base::key_equal](#) [key_equal](#)
- typedef [_Base::key_type](#) [key_type](#)
- typedef [__gnu_debug::Safe_local_iterator](#)< [_Base_local_iterator](#), [unordered_multiset](#) > [local_iterator](#)
- using [node_type](#) = typename [_Base::node_type](#)
- typedef [_Base::pointer](#) [pointer](#)
- typedef [_Base::reference](#) [reference](#)
- typedef [_Base::size_type](#) [size_type](#)
- typedef [_Base::value_type](#) [value_type](#)

Public Member Functions

- [unordered_multiset](#) ([_Base_ref](#) __x)
- template<typename [_InputIterator](#) >
[unordered_multiset](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, [size_type](#) __n, const [allocator_type](#) &__a)
- template<typename [_InputIterator](#) >
[unordered_multiset](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- template<typename [_InputIterator](#) >
[unordered_multiset](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=key_equal(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_multiset](#) (const [allocator_type](#) &__a)
- [unordered_multiset](#) (const [unordered_multiset](#) &)=default
- [unordered_multiset](#) (const [unordered_multiset](#) &__uset, const [allocator_type](#) &__a)
- [unordered_multiset](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n, const [allocator_type](#) &__a)
- [unordered_multiset](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_multiset](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=key_equal(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_multiset](#) ([size_type](#) __n, const [allocator_type](#) &__a)
- [unordered_multiset](#) ([size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_multiset](#) ([size_type](#) __n, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=key_equal(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_multiset](#) ([unordered_multiset](#) &&)=default
- [unordered_multiset](#) ([unordered_multiset](#) &&__uset, const [allocator_type](#) &__a) noexcept(noexcept([_Base](#)(std::move(↵__uset), __a)))
- const [_Base](#) & [_M_base](#) () const noexcept
- [_Base](#) & [_M_base](#) () noexcept
- [const_iterator](#) [begin](#) () const noexcept
- [iterator](#) [begin](#) () noexcept
- [local_iterator](#) [begin](#) ([size_type](#) __b)
- [const_local_iterator](#) [begin](#) ([size_type](#) __b) const
- [size_type](#) [bucket_size](#) ([size_type](#) __b) const
- [const_iterator](#) [cbegin](#) () const noexcept
- [const_local_iterator](#) [cbegin](#) ([size_type](#) __b) const
- [const_iterator](#) [cend](#) () const noexcept
- [const_local_iterator](#) [cend](#) ([size_type](#) __b) const
- void [clear](#) () noexcept
- template<typename... [_Args](#)>
[iterator](#) [emplace](#) ([_Args](#) &&... __args)
- template<typename... [_Args](#)>
[iterator](#) [emplace_hint](#) ([const_iterator](#) __hint, [_Args](#) &&... __args)

- `const_iterator` **end** () const noexcept
- `iterator` **end** () noexcept
- `local_iterator` **end** (size_type __b)
- `const_local_iterator` **end** (size_type __b) const
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
`std::pair< iterator, iterator >` **equal_range** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
`std::pair< const_iterator, const_iterator >` **equal_range** (const _Kt &__k) const
- `std::pair< iterator, iterator >` **equal_range** (const key_type &__key)
- `std::pair< const_iterator, const_iterator >` **equal_range** (const key_type &__key) const
- `_Base_iterator` **erase** (_Base_const_iterator __it)
- size_type **erase** (const key_type &__key)
- `iterator` **erase** (const_iterator __first, const_iterator __last)
- `iterator` **erase** (const_iterator __it)
- `iterator` **erase** (iterator __it)
- node_type **extract** (const key_type &__key)
- node_type **extract** (const_iterator __position)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
`iterator` **find** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
`const_iterator` **find** (const _Kt &__k) const
- `iterator` **find** (const key_type &__key)
- `const_iterator` **find** (const key_type &__key) const
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- `iterator` **insert** (const value_type &__obj)
- `iterator` **insert** (const_iterator __hint, const value_type &__obj)
- `iterator` **insert** (const_iterator __hint, node_type &&__nh)
- `iterator` **insert** (const_iterator __hint, value_type &&__obj)
- `iterator` **insert** (node_type &&__nh)
- void **insert** (std::initializer_list< value_type > __l)
- `iterator` **insert** (value_type &&__obj)
- float **max_load_factor** () const noexcept
- void **max_load_factor** (float __f)
- template<typename _H2, typename _P2 >
void **merge** (unordered_multiset< _Value, _H2, _P2, _Alloc > &&__source)
- template<typename _H2, typename _P2 >
void **merge** (unordered_multiset< _Value, _H2, _P2, _Alloc > &__source)
- template<typename _H2, typename _P2 >
void **merge** (unordered_set< _Value, _H2, _P2, _Alloc > &&__source)
- template<typename _H2, typename _P2 >
void **merge** (unordered_set< _Value, _H2, _P2, _Alloc > &__source)
- unordered_multiset & **operator=** (const unordered_multiset &)=default
- unordered_multiset & **operator=** (initializer_list< value_type > __l)
- unordered_multiset & **operator=** (unordered_multiset &&)=default
- void **swap** (unordered_multiset &__x) noexcept(noexcept(declval< _Base & >().swap(__x)))

Protected Member Functions

- void **_M_swap** (_Safe_container &__x) noexcept

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT >`
`class ::__gnu_debug:: Safe_iterator`
- `template<typename _ItT, typename _SeqT >`
`class ::__gnu_debug:: Safe_local_iterator`

6.1045.1 Detailed Description

```
template<typename _Value, typename _Hash = std::hash<_Value>, typename _Pred = std::equal_to<_Value>,
typename _Alloc = std::allocator<_Value>>
class std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >
```

Class std::unordered_multiset with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [debug/unordered_set](#)

6.1046 std::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference

```
#include <unordered_set.h>
```

Public Types

- using **node_type** = typename _Hashtable::node_type
- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)
- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- [unordered_multiset](#) ()=default
- `template<typename _InputIterator >`
unordered_multiset (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [allocator_type](#) &__a)
- `template<typename _InputIterator >`
unordered_multiset (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)

- `template<typename _InputIterator >`
`unordered_multiset` (`_InputIterator` __first, `_InputIterator` __last, `size_type` __n=0, const `hasher` &__hf=`hasher`(),
const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_multiset` (const `allocator_type` &__a)
- `unordered_multiset` (const `unordered_multiset` &)=default
- `unordered_multiset` (const `unordered_multiset` &__umset, const `allocator_type` &__a)
- `unordered_multiset` (`initializer_list`< `value_type` > __l, `size_type` __n, const `allocator_type` &__a)
- `unordered_multiset` (`initializer_list`< `value_type` > __l, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `unordered_multiset` (`initializer_list`< `value_type` > __l, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_multiset` (`size_type` __n, const `allocator_type` &__a)
- `unordered_multiset` (`size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `unordered_multiset` (`size_type` __n, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_multiset` (`unordered_multiset` &&)=default
- `unordered_multiset` (`unordered_multiset` &&__umset, const `allocator_type` &__a) noexcept(noexcept(←
`Hashtable`(`std::move`(__umset._M_h), __a)))
- `size_type bucket` (const `key_type` &__key) const
- `size_type bucket_count` () const noexcept
- `size_type bucket_size` (`size_type` __n) const
- `const_iterator cbegin` () const noexcept
- `const_iterator cend` () const noexcept
- void `clear` () noexcept
- `template<typename... _Args>`
`iterator emplace` (`_Args` &&... __args)
- `template<typename... _Args>`
`iterator emplace_hint` (const `iterator` __pos, `_Args` &&... __args)
- bool `empty` () const noexcept
- `size_type erase` (const `key_type` &__x)
- `iterator erase` (const `iterator` __first, const `iterator` __last)
- `node_type extract` (const `key_type` &__key)
- `node_type extract` (const `iterator` __pos)
- `allocator_type get_allocator` () const noexcept
- `hasher hash_function` () const
- `template<typename _InputIterator >`
void `insert` (`_InputIterator` __first, `_InputIterator` __last)
- `iterator insert` (const `iterator` __hint, `node_type` &&__nh)
- void `insert` (`initializer_list`< `value_type` > __l)
- `iterator insert` (`node_type` &&__nh)
- `key_equal key_eq` () const
- float `load_factor` () const noexcept
- `size_type max_bucket_count` () const noexcept
- float `max_load_factor` () const noexcept
- void `max_load_factor` (float __z)
- `size_type max_size` () const noexcept
- `template<typename _H2, typename _P2 >`
void `merge` (`unordered_multiset`< `_Value`, `_H2`, `_P2`, `_Alloc` > &&__source)
- `template<typename _H2, typename _P2 >`
void `merge` (`unordered_multiset`< `_Value`, `_H2`, `_P2`, `_Alloc` > &__source)
- `template<typename _H2, typename _P2 >`
void `merge` (`unordered_set`< `_Value`, `_H2`, `_P2`, `_Alloc` > &&__source)

- template<typename _H2, typename _P2 >
void **merge** (unordered_set< _Value, _H2, _P2, _Alloc > &__source)
- unordered_multiset & operator= (const unordered_multiset &)=default
- unordered_multiset & operator= (initializer_list< value_type > __l)
- unordered_multiset & operator= (unordered_multiset &&)=default
- void **rehash** (size_type __n)
- void **reserve** (size_type __n)
- size_type **size** () const noexcept
- void **swap** (unordered_multiset &__x) noexcept(noexcept(_M_h.swap(__x._M_h)))

- **iterator begin** () noexcept
- **const_iterator begin** () const noexcept

- **iterator end** () noexcept
- **const_iterator end** () const noexcept

- **iterator insert** (const value_type &__x)
- **iterator insert** (value_type &&__x)

- **iterator insert** (const_iterator __hint, const value_type &__x)
- **iterator insert** (const_iterator __hint, value_type &&__x)

- **iterator erase** (const_iterator __position)
- **iterator erase** (iterator __position)

- **iterator find** (const key_type &__x)
- template<typename _Kt >
auto **find** (const _Kt &__x) -> decltype(_M_h._M_find_tr(__x))
- **const_iterator find** (const key_type &__x) const
- template<typename _Kt >
auto **find** (const _Kt &__x) const -> decltype(_M_h._M_find_tr(__x))

- **size_type count** (const key_type &__x) const
- template<typename _Kt >
auto **count** (const _Kt &__x) const -> decltype(_M_h._M_count_tr(__x))

- bool **contains** (const key_type &__x) const
- template<typename _Kt >
auto **contains** (const _Kt &__x) const -> decltype(_M_h._M_find_tr(__x), void(), true)

- `std::pair< iterator, iterator > equal_range (const key_type &__x)`
- `template<typename _Kt >`
`auto equal_range (const _Kt &__x) -> decltype(_M_h._M_equal_range_tr(__x))`
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__x) const`
- `template<typename _Kt >`
`auto equal_range (const _Kt &__x) const -> decltype(_M_h._M_equal_range_tr(__x))`
- `local_iterator begin (size_type __n)`
- `const_local_iterator begin (size_type __n) const`
- `const_local_iterator cbegin (size_type __n) const`
- `local_iterator end (size_type __n)`
- `const_local_iterator end (size_type __n) const`
- `const_local_iterator cend (size_type __n) const`

Friends

- `template<typename _Value1, typename _Hash1, typename _Pred1, typename _Alloc1 >`
`bool operator== (const unordered_multiset< _Value1, _Hash1, _Pred1, _Alloc1 > &, const unordered_multiset< _Value1, _Hash1, _Pred1, _Alloc1 > &)`
- `template<typename, typename, typename >`
`class std::_Hash_merge_helper`

6.1046.1 Detailed Description

```
template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>, type-
name _Alloc = allocator<_Value>>
class std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >
```

A standard container composed of equivalent keys (possibly containing multiple of each key value) in which the elements' keys are the elements themselves.

Template Parameters

<code>_Value</code>	Type of key objects.
<code>_Hash</code>	Hashing function object type, defaults to <code>hash<_Value></code> .
<code>_Pred</code>	Predicate function object type, defaults to <code>equal_to<_Value></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a [container](#), and [unordered associative container](#)

Base is `_Hashtable`, dispatched at compile time via template alias `__umset_hashtable`.

6.1046.2 Member Typedef Documentation

allocator_type

```
template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::allocator_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >↵
::allocator_type
```

Public typedefs.

const_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >↵
::const_iterator
Iterator-related typedefs.
```

const_local_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
>::const_local_iterator
Iterator-related typedefs.
```

const_pointer

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_pointer std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::const↵
_pointer
Iterator-related typedefs.
```

const_reference

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_reference std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >↵
::const_reference
Iterator-related typedefs.
```

difference_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::difference_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >↵
::difference_type
Iterator-related typedefs.
```

hasher

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::hasher std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::hasher
Public typedefs.
```

iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::iterator
Iterator-related typedefs.
```


key_equal

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::key_equal std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::key_equal
Public typedefs.
```

key_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::key_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::key_type
Public typedefs.
```

local_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >↵
::local_iterator
Iterator-related typedefs.
```

pointer

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::pointer std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::pointer
Iterator-related typedefs.
```

reference

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::reference std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::reference
Iterator-related typedefs.
```

size_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::size_type
Iterator-related typedefs.
```

value_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::value_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::value_↵
type
Public typedefs.
```

6.1046.3 Constructor & Destructor Documentation**unordered_multiset()** [1/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
```

```
typename _Alloc = allocator<_Value>>
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset ( ) [default]
Default constructor.
```

unordered_multiset() [2/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    size_type __n,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

unordered_multiset() [3/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _InputIterator >
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    _InputIterator __first,
    _InputIterator __last,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multiset consisting of copies of the elements from [`__first`,`__last`). This is linear in N (where N is distance(`__first`,`__last`)).

unordered_multiset() [4/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    const unordered_multiset< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy constructor.

unordered_multiset() [5/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    unordered_multiset< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move constructor.

unordered_multiset() [6/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    initializer_list< value_type > __l,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_multiset from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multiset consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

unordered_multiset() [7/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
    const allocator_type & __a ) [inline], [explicit]
```

Creates an unordered_multiset with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

6.1046.4 Member Function Documentation

begin() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::begin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multiset.

begin() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multiset.

begin() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::begin (
    size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

begin() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::begin (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

bucket_count()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::bucket_count ( ) const [inline],
[noexcept]
```

Returns the number of buckets of the unordered_multiset.

cbegin() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_multiset`.

cbegin() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cbegin (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

cend() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_multiset`.

Referenced by `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert()`.

cend() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cend (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

clear()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
```

```
typename _Alloc = allocator<_Value>>
```

```
void std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in an unordered_multiset.

Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value> ,
```


```
typename _Alloc = allocator<_Value>>
```

```
template<typename _Kt >
```

```
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::contains (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x), void(), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

 __x	Key of elements to be located.
--	--------------------------------

Returns

True if there is any element with the specified key.

contains() [2/2]


```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value> ,
```

```
typename _Alloc = allocator<_Value>>
```

```
bool std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

 __x	Key of elements to be located.
--	--------------------------------

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value> ,
```

```
typename _Alloc = allocator<_Value>>
```

```
template<typename _Kt >
```

```
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::count (
    const _Kt & __x ) const -> decltype(_M_h._M_count_tr(__x)) [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

count() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::count (
    const key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

emplace()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename... _Args>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::emplace (
    _Args &&... __args ) [inline]
```

Builds and insert an element into the unordered_multiset.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

emplace_hint()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename... _Args>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Inserts an element into the `unordered_multiset`.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant time.

empty()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
bool std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the `unordered_multiset` is empty.

end() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_multiset`.

end() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end ( ) [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_multiset`.

Referenced by `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::extract()`.

end() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end (
    size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__↵</code>	The bucket index.
<code>__n</code>	

Returns

A read-only local iterator.

end() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

equal_range() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __x ) -> decltype(_M_h._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __x ) const -> decltype(_M_h._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< iterator, iterator > std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::equal_range (
    const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code> <code>_X</code>	Key to be located.
------------------------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

equal_range() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< const_iterator, const_iterator > std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
>::equal_range (
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code> <code>_X</code>	Key to be located.
------------------------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

erase() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>_↔</code> <code>_X</code>	Key of element to be erased.
------------------------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [*__first*,*__last*) range of elements from an unordered_multiset.

Parameters

<i>__first</i>	Iterator pointing to the start of the range to be erased.
<i>__last</i>	Iterator pointing to the end of the range to be erased.

Returns

The iterator *__last*.

This function erases a sequence of elements from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from an unordered_multiset.

Parameters

<i>__position</i>	An iterator pointing to the element to be erased.
-------------------	---

Returns

An iterator pointing to the element immediately following *__position* prior to the element being erased. If no such element exists, end() is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from an unordered_multiset.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

extract() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
node_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::extract (
    const key_type & __key ) [inline]
```

Extract a node.

extract() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
node_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end()`.

find() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) -> decltype(_M_h._M_find_tr(__x)) [inline]
```

Tries to locate an element in an unordered_multiset.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or `end()` if not found.


This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::find (
    const _Kt & __x ) const -> decltype(_M_h._M_find_tr(__x))    [inline]
```

Tries to locate an element in an unordered_multiset.

Parameters

 __x	Element to be located.
--	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.


This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::find (
    const key_type & __x )    [inline]
```

Tries to locate an element in an unordered_multiset.

Parameters

 __x	Element to be located.
--	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.


This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) const    [inline]
```

Tries to locate an element in an unordered_multiset.

Parameters

 __x	Element to be located.
--	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

get_allocator()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
allocator_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::get_allocator ( ) const
[inline], [noexcept]
```

Returns the allocator object used by the unordered_multiset.

hash_function()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
hasher std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::hash_function ( ) const [inline]
```

Returns the hash functor object with which the unordered_multiset was constructed.

insert() [1/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _InputIterator >
void std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that inserts a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Inserts an element into the unordered_multiset.

Parameters

<code>__x</code>	Element to be inserted.
------------------	-------------------------

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

insert() [3/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    const value_type & __x ) [inline]
```

Inserts an element into the unordered_multiset.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires amortized constant.

insert() [4/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [5/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    value_type && __x ) [inline]
```

Inserts an element into the unordered_multiset.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires amortized constant.

References [std::move\(\)](#).

insert() [6/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]
```

Inserts a list of elements into the unordered_multiset.

Parameters

↔	A std::initializer_list<value_type> of elements to be inserted.
↔	
↔	
↔	
l	

Complexity similar to that of the range constructor.

insert() [7/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cend\(\)](#), and [std::move\(\)](#).

insert() [8/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
    value_type && __x ) [inline]
```

Inserts an element into the unordered_multiset.

Parameters

↔	Element to be inserted.
x	

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

References [std::move\(\)](#).

key_eq()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
key_equal std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::key_eq ( ) const [inline]
```

Returns the key comparison object with which the unordered_multiset was constructed.

load_factor()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
float std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::load_factor ( ) const [inline],
[noexcept]
```

Returns the average number of elements per bucket.

max_bucket_count()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::max_bucket_count ( ) const
[inline], [noexcept]
```

Returns the maximum number of buckets of the unordered_multiset.

max_load_factor() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
float std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::max_load_factor ( ) const [inline],
[noexcept]
```

Returns a positive number that the unordered_multiset tries to keep the load factor less than or equal to.


max_load_factor() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
void std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::max_load_factor (
float __z ) [inline]
```

Change the unordered_multiset maximum load factor.

Parameters

 <code>__z</code>	The new maximum load factor.
--	------------------------------

max_size()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::max_size ( ) const [inline],
[noexcept]
```

Returns the maximum size of the unordered_multiset.

operator=() [1/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_multiset & std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::operator= (
    const unordered_multiset< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy assignment operator.

operator=() [2/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_multiset & std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Unordered_multiset list assignment operator.

Parameters

↩	An initializer_list.
↩	
↩	
↩	
/	

This function fills an unordered_multiset with copies of the elements in the initializer list __l.

Note that the assignment completely changes the unordered_multiset and that the resulting unordered_multiset's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_multiset & std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::operator= (
    unordered_multiset< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move assignment operator.

rehash()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::rehash (
    size_type __n ) [inline]
```

May rehash the unordered_multiset.

Parameters

↩	The new number of buckets.
n	

Rehash will occur only if the new number of buckets respect the unordered_multiset maximum load factor.

reserve()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc >::reserve (
    size_type __n ) [inline]
```

Prepare the unordered_multiset for a specified number of elements.

Parameters

\leftrightarrow	Number of elements required.
<code>__n</code>	

Same as `rehash(ceil(n / max_load_factor()))`.

size()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc >::size ( ) const [inline],
[noexcept]
```

Returns the size of the unordered_multiset.

swap()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc >::swap (
    unordered_multiset<_Value, _Hash, _Pred, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another unordered_multiset.

Parameters

\leftrightarrow	An unordered_multiset of the same element and allocator types.
<code>__x</code>	

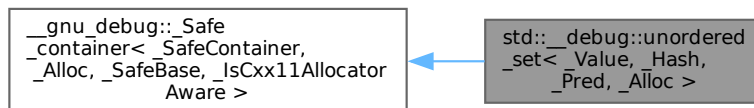
This exchanges the elements between two sets in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

The documentation for this class was generated from the following file:

- [unordered_set.h](#)

6.1047 std::__debug::unordered_set<_Value, _Hash, _Pred, _Alloc > Class Template Reference

Inheritance diagram for `std::__debug::unordered_set<_Value, _Hash, _Pred, _Alloc >`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< _Base_const_iterator, unordered_set >` **const_iterator**
- typedef `__gnu_debug::Safe_local_iterator< _Base_const_local_iterator, unordered_set >` **const_local_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `_Base::difference_type` **difference_type**
- typedef `_Base::hasher` **hasher**
- using **insert_return_type** = `_Node_insert_return< iterator, node_type >`
- typedef `__gnu_debug::Safe_iterator< _Base_iterator, unordered_set >` **iterator**
- typedef `_Base::key_equal` **key_equal**
- typedef `_Base::key_type` **key_type**
- typedef `__gnu_debug::Safe_local_iterator< _Base_local_iterator, unordered_set >` **local_iterator**
- using **node_type** = `typename _Base::node_type`
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `_Base::size_type` **size_type**
- typedef `_Base::value_type` **value_type**

Public Member Functions

- **unordered_set** (`_Base_ref __x`)
- `template<typename _InputIterator >`
unordered_set (`_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a`)
- `template<typename _InputIterator >`
unordered_set (`_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a`)
- `template<typename _InputIterator >`
unordered_set (`_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type()`)
- **unordered_set** (`const allocator_type &__a`)
- **unordered_set** (`const unordered_set &`)=default
- **unordered_set** (`const unordered_set &__uset, const allocator_type &__a`)
- **unordered_set** (`initializer_list< value_type > __l, size_type __n, const allocator_type &__a`)
- **unordered_set** (`initializer_list< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a`)
- **unordered_set** (`initializer_list< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type()`)
- **unordered_set** (`size_type __n, const allocator_type &__a`)
- **unordered_set** (`size_type __n, const hasher &__hf, const allocator_type &__a`)
- **unordered_set** (`size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type()`)
- **unordered_set** (`unordered_set &&`)=default
- **unordered_set** (`unordered_set &&__uset, const allocator_type &__a`) noexcept(noexcept(`_Base(std::move(__uset), __a)`))
- `const _Base & _M_base ()` const noexcept
- `_Base & _M_base ()` noexcept
- **const_iterator begin** () const noexcept
- **iterator begin** () noexcept
- **local_iterator begin** (`size_type __b`)
- **const_local_iterator begin** (`size_type __b`) const

- size_type **bucket_size** (size_type __b) const
- **const_iterator** **cbegin** () const noexcept
- **const_local_iterator** **cbegin** (size_type __b) const
- **const_iterator** **cend** () const noexcept
- **const_local_iterator** **cend** (size_type __b) const
- void **clear** () noexcept
- template<typename... _Args>
std::pair< **iterator**, bool > **emplace** (_Args &&... __args)
- template<typename... _Args>
iterator **emplace_hint** (**const_iterator** __hint, _Args &&... __args)
- **const_iterator** **end** () const noexcept
- **iterator** **end** () noexcept
- **local_iterator** **end** (size_type __b)
- **const_local_iterator** **end** (size_type __b) const
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
std::pair< **iterator**, **iterator** > **equal_range** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
std::pair< **const_iterator**, **const_iterator** > **equal_range** (const _Kt &__k) const
- std::pair< **iterator**, **iterator** > **equal_range** (const key_type &__key)
- std::pair< **const_iterator**, **const_iterator** > **equal_range** (const key_type &__key) const
- **_Base_iterator** **erase** (_Base_const_iterator __it)
- size_type **erase** (const key_type &__key)
- **iterator** **erase** (**const_iterator** __first, **const_iterator** __last)
- **iterator** **erase** (**const_iterator** __it)
- **iterator** **erase** (**iterator** __it)
- node_type **extract** (const key_type &__key)
- node_type **extract** (**const_iterator** __position)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
iterator **find** (const _Kt &__k)
- template<typename _Kt, typename = std::__has_is_transparent_t<_Hash, _Kt>, typename = std::__has_is_transparent_t<_Pred, _Kt>>
const_iterator **find** (const _Kt &__k) const
- **iterator** **find** (const key_type &__key)
- **const_iterator** **find** (const key_type &__key) const
- template<typename _InputIterator>
void **insert** (_InputIterator __first, _InputIterator __last)
- std::pair< **iterator**, bool > **insert** (const value_type &__obj)
- **iterator** **insert** (**const_iterator** __hint, const value_type &__obj)
- **iterator** **insert** (**const_iterator** __hint, node_type &&__nh)
- **iterator** **insert** (**const_iterator** __hint, value_type &&__obj)
- **insert_return_type** **insert** (node_type &&__nh)
- void **insert** (std::initializer_list< value_type > __l)
- std::pair< **iterator**, bool > **insert** (value_type &&__obj)
- float **max_load_factor** () const noexcept
- void **max_load_factor** (float __f)
- template<typename _H2, typename _P2>
void **merge** (unordered_multiset< _Value, _H2, _P2, _Alloc > &&__source)
- template<typename _H2, typename _P2>
void **merge** (unordered_multiset< _Value, _H2, _P2, _Alloc > &__source)
- template<typename _H2, typename _P2>
void **merge** (unordered_set< _Value, _H2, _P2, _Alloc > &&__source)
- template<typename _H2, typename _P2>
void **merge** (unordered_set< _Value, _H2, _P2, _Alloc > &__source)

- `unordered_set` & `operator=` (const `unordered_set` &)=default
- `unordered_set` & `operator=` (initializer_list< value_type > __l)
- `unordered_set` & `operator=` (`unordered_set` &&)=default
- void `swap` (`unordered_set` &__x) noexcept(noexcept(declval< _Base & >().swap(__x)))

Protected Member Functions

- void `_M_swap` (_Safe_container &__x) noexcept

Friends

- template<typename _ItT, typename _SeqT, typename _CatT >
class ::`__gnu_debug::_Safe_iterator`
- template<typename _ItT, typename _SeqT >
class ::`__gnu_debug::_Safe_local_iterator`

6.1047.1 Detailed Description

template<typename _Value, typename _Hash = std::hash<_Value>, typename _Pred = std::equal_to<_Value>,
typename _Alloc = std::allocator<_Value>>
class std::`__debug::unordered_set`< _Value, _Hash, _Pred, _Alloc >

Class std::unordered_set with safety/checking/debug instrumentation.

The documentation for this class was generated from the following file:

- [debug/unordered_set](#)

6.1048 std::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference

```
#include <unordered_set.h>
```

Public Types

- using `insert_return_type` = typename _Hashtable::insert_return_type
- using `node_type` = typename _Hashtable::node_type

- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)

- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- [unordered_set](#) ()=default
- [template<typename _InputIterator > unordered_set](#) (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [allocator_type](#) &__a)
- [template<typename _InputIterator > unordered_set](#) (_InputIterator __first, _InputIterator __last, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [template<typename _InputIterator > unordered_set](#) (_InputIterator __first, _InputIterator __last, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_set](#) (const [allocator_type](#) &__a)
- [unordered_set](#) (const [unordered_set](#) &)=default
- [unordered_set](#) (const [unordered_set](#) &__uset, const [allocator_type](#) &__a)
- [unordered_set](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n, const [allocator_type](#) &__a)
- [unordered_set](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_set](#) ([initializer_list](#)< [value_type](#) > __l, [size_type](#) __n=0, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_set](#) ([size_type](#) __n, const [allocator_type](#) &__a)
- [unordered_set](#) ([size_type](#) __n, const [hasher](#) &__hf, const [allocator_type](#) &__a)
- [unordered_set](#) ([size_type](#) __n, const [hasher](#) &__hf=[hasher](#)(), const [key_equal](#) &__eq=[key_equal](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- [unordered_set](#) ([unordered_set](#) &&)=default
- [unordered_set](#) ([unordered_set](#) &&__uset, const [allocator_type](#) &__a) noexcept(noexcept(_Hashtable([std::move](#)(← __uset._M_h), __a)))
- [size_type bucket](#) (const [key_type](#) &__key) const
- [size_type bucket_count](#) () const noexcept
- [size_type bucket_size](#) ([size_type](#) __n) const
- [const_iterator cbegin](#) () const noexcept
- [const_iterator cend](#) () const noexcept
- void [clear](#) () noexcept
- [template<typename... _Args> std::pair< iterator, bool > emplace](#) (_Args &&... __args)
- [template<typename... _Args> iterator emplace_hint](#) (const [iterator](#) __pos, _Args &&... __args)
- bool [empty](#) () const noexcept
- [size_type erase](#) (const [key_type](#) &__x)
- [iterator erase](#) (const [iterator](#) __first, const [iterator](#) __last)
- [node_type extract](#) (const [key_type](#) &__key)
- [node_type extract](#) (const [iterator](#) __pos)
- [allocator_type get_allocator](#) () const noexcept
- [hasher hash_function](#) () const
- [template<typename _InputIterator > void insert](#) (_InputIterator __first, _InputIterator __last)
- [iterator insert](#) (const [iterator](#), [node_type](#) &&__nh)
- void [insert](#) ([initializer_list](#)< [value_type](#) > __l)
- [insert_return_type insert](#) ([node_type](#) &&__nh)
- [key_equal key_eq](#) () const
- float [load_factor](#) () const noexcept
- [size_type max_bucket_count](#) () const noexcept
- float [max_load_factor](#) () const noexcept
- void [max_load_factor](#) (float __z)

- `size_type max_size ()` const noexcept
 - `template<typename _H2, typename _P2 >`
`void merge (unordered_multiset< _Value, _H2, _P2, _Alloc > &&__source)`
 - `template<typename _H2, typename _P2 >`
`void merge (unordered_multiset< _Value, _H2, _P2, _Alloc > &__source)`
 - `template<typename _H2, typename _P2 >`
`void merge (unordered_set< _Value, _H2, _P2, _Alloc > &&__source)`
 - `template<typename _H2, typename _P2 >`
`void merge (unordered_set< _Value, _H2, _P2, _Alloc > &__source)`
 - `unordered_set & operator= (const unordered_set &)=default`
 - `unordered_set & operator= (initializer_list< value_type > __l)`
 - `unordered_set & operator= (unordered_set &&)=default`
 - `void rehash (size_type __n)`
 - `void reserve (size_type __n)`
 - `size_type size ()` const noexcept
 - `void swap (unordered_set &__x)` noexcept(noexcept(_M_h.swap(__x._M_h)))
-
- `iterator begin ()` noexcept
 - `const_iterator begin ()` const noexcept
-
- `iterator end ()` noexcept
 - `const_iterator end ()` const noexcept
-
- `std::pair< iterator, bool > insert (const value_type &__x)`
 - `std::pair< iterator, bool > insert (value_type &&__x)`
-
- `iterator insert (const_iterator __hint, const value_type &__x)`
 - `iterator insert (const_iterator __hint, value_type &&__x)`
-
- `iterator erase (const_iterator __position)`
 - `iterator erase (iterator __position)`
-
- `iterator find (const key_type &__x)`
 - `template<typename _Kt >`
`auto find (const _Kt &__k) -> decltype(_M_h._M_find_tr(__k))`
 - `const_iterator find (const key_type &__x) const`
 - `template<typename _Kt >`
`auto find (const _Kt &__k) const -> decltype(_M_h._M_find_tr(__k))`
-
- `size_type count (const key_type &__x) const`
 - `template<typename _Kt >`
`auto count (const _Kt &__k) const -> decltype(_M_h._M_count_tr(__k))`

- bool `contains` (const `key_type` &__x) const
- template<typename `_Kt` >
auto `contains` (const `_Kt` &__k) const -> decltype(_M_h._M_find_tr(__k), void(), true)
- `std::pair`< `iterator`, `iterator` > `equal_range` (const `key_type` &__x)
- template<typename `_Kt` >
auto `equal_range` (const `_Kt` &__k) -> decltype(_M_h._M_equal_range_tr(__k))
- `std::pair`< `const_iterator`, `const_iterator` > `equal_range` (const `key_type` &__x) const
- template<typename `_Kt` >
auto `equal_range` (const `_Kt` &__k) const -> decltype(_M_h._M_equal_range_tr(__k))
- `local_iterator` `begin` (`size_type` __n)
- `const_local_iterator` `begin` (`size_type` __n) const
- `const_local_iterator` `cbegin` (`size_type` __n) const
- `local_iterator` `end` (`size_type` __n)
- `const_local_iterator` `end` (`size_type` __n) const
- `const_local_iterator` `cend` (`size_type` __n) const

Friends

- template<typename `_Value1` , typename `_Hash1` , typename `_Pred1` , typename `_Alloc1` >
bool **operator==** (const `unordered_set`< `_Value1`, `_Hash1`, `_Pred1`, `_Alloc1` > &, const `unordered_set`< `_Value1`, `_Hash1`, `_Pred1`, `_Alloc1` > &)
- template<typename , typename , typename >
class **std::_Hash_merge_helper**

6.1048.1 Detailed Description

template<typename `_Value`, typename `_Hash` = hash<_Value>, typename `_Pred` = equal_to<_Value>, type-name `_Alloc` = allocator<_Value>>

class std::unordered_set< _Value, _Hash, _Pred, _Alloc >

A standard container composed of unique keys (containing at most one of each key value) in which the elements' keys are the elements themselves.

Template Parameters

<code>_Value</code>	Type of key objects.
<code>_Hash</code>	Hashing function object type, defaults to hash<_Value>.
<code>_Pred</code>	Predicate function object type, defaults to equal_to<_Value>.
<code>_Alloc</code>	Allocator type, defaults to allocator<_Key>.

Meets the requirements of a `container`, and `unordered associative container`

Base is `_Hashtable`, dispatched at compile time via template alias `__uset_hashtable`.

6.1048.2 Member Typedef Documentation

allocator_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::allocator_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::allocator←
_type
```

Public typedefs.

const_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::const←
iterator
```

Iterator-related typedefs.

const_local_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >←
::const_local_iterator
```

Iterator-related typedefs.

const_pointer

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_pointer std::unordered_set< _Value, _Hash, _Pred, _Alloc >::const←
pointer
```

Iterator-related typedefs.

const_reference

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::const_reference std::unordered_set< _Value, _Hash, _Pred, _Alloc >::const←
reference
```

Iterator-related typedefs.

difference_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::difference_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::difference←
_type
```

Iterator-related typedefs.

hasher

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::hasher std::unordered_set< _Value, _Hash, _Pred, _Alloc >::hasher
```

Public typedefs.

iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::iterator
Iterator-related typedefs.
```

key_equal

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::key_equal std::unordered_set< _Value, _Hash, _Pred, _Alloc >::key_equal
Public typedefs.
```

key_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::key_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::key_type
Public typedefs.
```

local_iterator

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::local_↵
iterator
Iterator-related typedefs.
```

pointer

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::pointer std::unordered_set< _Value, _Hash, _Pred, _Alloc >::pointer
Iterator-related typedefs.
```

reference

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::reference std::unordered_set< _Value, _Hash, _Pred, _Alloc >::reference
Iterator-related typedefs.
```

size_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::size_type
Iterator-related typedefs.
```

value_type

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
typedef _Hashtable::value_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::value_type
Public typedefs.
```

6.1048.3 Constructor & Destructor Documentation

unordered_set() [1/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set ( ) [default]
Default constructor.
```

unordered_set() [2/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    size_type __n,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline], [explicit]
```

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

unordered_set() [3/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _InputIterator >
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    _InputIterator __first,
    _InputIterator __last,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_set consisting of copies of the elements from [`__first`,`__last`). This is linear in N (where N is distance(`__first`,`__last`)).

unordered_set() [4/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    const unordered_set< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy constructor.

unordered_set() [5/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    unordered_set< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move constructor.

unordered_set() [6/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    const allocator_type & __a ) [inline], [explicit]
```

Creates an unordered_set with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

unordered_set() [7/7]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set (
    initializer_list< value_type > __l,
    size_type __n = 0,
    const hasher & __hf = hasher(),
    const key_equal & __eq1 = key_equal(),
    const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_set from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eq1</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_set consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

6.1048.4 Member Function Documentation

begin() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::begin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_set`.

begin() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::begin ( ) [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_set`.

begin() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::begin (
    size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

begin() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::begin (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

bucket_count()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
```

```
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::bucket_count ( ) const [inline],
[noexcept]
```

Returns the number of buckets of the unordered_set.

cbegin() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::cbegin ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_set.

cbegin() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::cbegin (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftarrow	The bucket index.
n	

Returns

A read-only local iterator.

cend() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::cend ( ) const [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_set.

cend() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::cend (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

\leftarrow	The bucket index.
n	

Returns

A read-only local iterator.

clear()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::clear ( ) [inline], [noexcept]
```

Erases all elements in an `unordered_set`. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

contains() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::contains (
    const _Kt & __k ) const -> decltype(_M_h._M_find_tr(__k), void()), true) [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__k</code>	Key of elements to be located.
<code>__x</code>	

Returns

True if there is any element with the specified key.

contains() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
bool std::unordered_set< _Value, _Hash, _Pred, _Alloc >::contains (
    const key_type & __x ) const [inline]
```

Finds whether an element with the given key exists.

Parameters

<code>__k</code>	Key of elements to be located.
<code>__x</code>	

Returns

True if there is any element with the specified key.

count() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::count (
    const _Kt & __k ) const -> decltype(_M_h._M_count_tr(__k)) [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

This function only makes sense for unordered_multisets; for unordered_set the result will either be 0 (not present) or 1 (present).

count() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::count (
    const key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

This function only makes sense for unordered_multisets; for unordered_set the result will either be 0 (not present) or 1 (present).

emplace()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename... _Args>
std::pair< iterator, bool > std::unordered_set< _Value, _Hash, _Pred, _Alloc >::emplace (
    _Args &&... __args ) [inline]
```

Attempts to build and insert an element into the unordered_set.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to build and insert an element into the unordered_set. An unordered_set relies on unique keys and thus an element is only inserted if it is not already present in the unordered_set.

Insertion requires amortized constant time.

emplace_hint()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename... _Args>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::emplace_hint (
    const_iterator __pos,
    _Args &&... __args ) [inline]
```

Attempts to insert an element into the `unordered_set`.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the element with key equivalent to the one generated from `__args` (may or may not be the element itself).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires amortized constant time.

empty()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
bool std::unordered_set< _Value, _Hash, _Pred, _Alloc >::empty ( ) const [inline], [noexcept]
```

Returns true if the `unordered_set` is empty.

end() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_set`.

end() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::end ( ) [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_set`.
Referenced by `std::unordered_set< _Value, _Hash, _Pred, _Alloc >::extract()`.

end() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::end (
    size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_↔$ $_n$	The bucket index.
----------------	-------------------

Returns

A read-only local iterator.

end() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_local_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::end (
    size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_↔$ $_n$	The bucket index.
----------------	-------------------

Returns

A read-only local iterator.

equal_range() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __k ) -> decltype(_M_h._M_equal_range_tr(__k)) [inline]
```

Finds a subsequence matching given key.

Parameters

$_↔$ $_x$	Key to be located.
----------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

equal_range() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::equal_range (
    const _Kt & __k ) const -> decltype(_M_h._M_equal_range_tr(__k)) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

equal_range() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< iterator, iterator > std::unordered_set< _Value, _Hash, _Pred, _Alloc >::equal_range
(
    const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

equal_range() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< const_iterator, const_iterator > std::unordered_set< _Value, _Hash, _Pred, _Alloc >↵
::equal_range (
    const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↵</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

erase() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase (
    const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an `unordered_set`. For an `unordered_set` the result of this function can only be 0 (not present) or 1 (present). Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase (
    const_iterator __first,
    const_iterator __last ) [inline]
```

Erases a [`__first`,`__last`) range of elements from an `unordered_set`.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an `unordered_set`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

erase() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase (
    const_iterator __position ) [inline]
```

Erases an element from an `unordered_set`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_set`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

`erase()` [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase (
    iterator __position ) [inline]
```

Erases an element from an `unordered_set`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_set`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

`extract()` [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
node_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::extract (
    const key_type & __key ) [inline]
```

Extract a node.

`extract()` [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
node_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::extract (
    const_iterator __pos ) [inline]
```

Extract a node.

References `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end()`.

find() [1/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::find (
    const _Kt & __k ) -> decltype(_M_h._M_find_tr(__k))    [inline]
```

Tries to locate an element in an `unordered_set`.

Parameters

<code>__k</code>	Element to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [2/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _Kt >
auto std::unordered_set< _Value, _Hash, _Pred, _Alloc >::find (
    const _Kt & __k ) const -> decltype(_M_h._M_find_tr(__k))    [inline]
```

Tries to locate an element in an `unordered_set`.

Parameters

<code>__k</code>	Element to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

find() [3/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::find (
    const key_type & __x )    [inline]
```

Tries to locate an element in an `unordered_set`.

Parameters

<code>__x</code>	Element to be located.
<code>__x</code>	

Returns

Iterator pointing to sought-after element, or end() if not found.


This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

find() [4/4]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
const_iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::find (
    const key_type & __x ) const [inline]
```

Tries to locate an element in an unordered_set.

Parameters

 <code>__x</code>	Element to be located.
--	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

get_allocator()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
allocator_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::get_allocator ( ) const [inline],
[noexcept]
```

Returns the allocator object used by the unordered_set.

hash_function()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
hasher std::unordered_set< _Value, _Hash, _Pred, _Alloc >::hash_function ( ) const [inline]
```

Returns the hash functor object with which the unordered_set was constructed.

insert() [1/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
template<typename _InputIterator >
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    _InputIterator __first,
    _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

insert() [2/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< iterator, bool > std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    const value_type & __x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

<code>__x</code>	Element to be inserted.
------------------	-------------------------

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to insert an element into the unordered_set. An unordered_set relies on unique keys and thus an element is only inserted if it is not already present in the unordered_set. Insertion requires amortized constant time.

insert() [3/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    const_iterator __hint,
    const value_type & __x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of __x (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument insert() does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant.

insert() [4/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
```

```
const_iterator __hint,
value_type && __x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of `__x` (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires amortized constant.

References [std::move\(\)](#).

insert() [5/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    const_iterator ,
    node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [6/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of elements into the unordered_set.

Parameters

<code>__l</code>	A std::initializer_list<value_type> of elements to be inserted.
------------------	---

Complexity similar to that of the range constructor.

insert() [7/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
insert_return_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
```

```
node_type && __nh ) [inline]
```

Re-insert an extracted node.

References [std::move\(\)](#).

insert() [8/8]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
std::pair< iterator, bool > std::unordered_set< _Value, _Hash, _Pred, _Alloc >::insert (
    value_type && __x ) [inline]
```

Attempts to insert an element into the `unordered_set`.

Parameters

<code>__x</code>	Element to be inserted.
------------------	-------------------------

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a `bool` that is true if the element was actually inserted.

This function attempts to insert an element into the `unordered_set`. An `unordered_set` relies on unique keys and thus an element is only inserted if it is not already present in the `unordered_set`.

Insertion requires amortized constant time.

References [std::move\(\)](#).

key_eq()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
key_equal std::unordered_set< _Value, _Hash, _Pred, _Alloc >::key_eq ( ) const [inline]
```

Returns the key comparison object with which the `unordered_set` was constructed.

load_factor()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
float std::unordered_set< _Value, _Hash, _Pred, _Alloc >::load_factor ( ) const [inline], [noexcept]
```

Returns the average number of elements per bucket.

max_bucket_count()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::max_bucket_count ( ) const [inline],
[noexcept]
```

Returns the maximum number of buckets of the `unordered_set`.

max_load_factor() [1/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
float std::unordered_set< _Value, _Hash, _Pred, _Alloc >::max_load_factor ( ) const [inline],
[noexcept]
```

Returns a positive number that the `unordered_set` tries to keep the load factor less than or equal to.

max_load_factor() [2/2]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::max_load_factor (
    float __z ) [inline]
```

Change the unordered_set maximum load factor.

Parameters

<code>__z</code>	The new maximum load factor.
------------------	------------------------------

max_size()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the unordered_set.

operator=() [1/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_set & std::unordered_set< _Value, _Hash, _Pred, _Alloc >::operator= (
    const unordered_set< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy assignment operator.

operator=() [2/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_set & std::unordered_set< _Value, _Hash, _Pred, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline]
```

Unordered_set list assignment operator.

Parameters

<code>__l</code>	An initializer_list.
------------------	----------------------

This function fills an unordered_set with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the unordered_set and that the resulting unordered_set's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
unordered_set & std::unordered_set< _Value, _Hash, _Pred, _Alloc >::operator= (
    unordered_set< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move assignment operator.

rehash()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::rehash (
    size_type __n ) [inline]
```

May rehash the unordered_set.

Parameters

<code>__n</code>	The new number of buckets.
------------------	----------------------------

Rehash will occur only if the new number of buckets respect the unordered_set maximum load factor.

reserve()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::reserve (
    size_type __n ) [inline]
```

Prepare the unordered_set for a specified number of elements.

Parameters

<code>__n</code>	Number of elements required.
------------------	------------------------------

Same as rehash(ceil(n / max_load_factor())).

size()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::size ( ) const [inline], [noexcept]
```

Returns the size of the unordered_set.

swap()

```
template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = equal_to<_Value>,
typename _Alloc = allocator<_Value>>
void std::unordered_set< _Value, _Hash, _Pred, _Alloc >::swap (
    unordered_set< _Value, _Hash, _Pred, _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another unordered_set.

Parameters

<code>__x</code>	An unordered_set of the same element and allocator types.
------------------	---

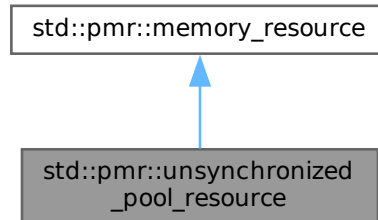
This exchanges the elements between two sets in constant time. Note that the global std::swap() function is specialized such that std::swap(s1,s2) will feed to this function.

The documentation for this class was generated from the following file:

- [unordered_set.h](#)

6.1049 std::pmr::unsynchronized_pool_resource Class Reference

Inheritance diagram for std::pmr::unsynchronized_pool_resource:



Public Member Functions

- **unsynchronized_pool_resource** (const [pool_options](#) &__opts)
- **unsynchronized_pool_resource** (const [pool_options](#) &__opts, [memory_resource](#) *__upstream)
- **unsynchronized_pool_resource** (const [unsynchronized_pool_resource](#) &)=delete
- **unsynchronized_pool_resource** ([memory_resource](#) *__upstream)
- void * **allocate** (size_t __bytes, size_t __alignment=_S_max_align)
- void **deallocate** (void * __p, size_t __bytes, size_t __alignment=_S_max_align)
- bool **is_equal** (const [memory_resource](#) &__other) const noexcept
- [unsynchronized_pool_resource](#) & **operator=** (const [unsynchronized_pool_resource](#) &)=delete
- [pool_options](#) **options** () const noexcept
- void **release** ()
- [memory_resource](#) * **upstream_resource** () const noexcept

Protected Member Functions

- void * **do_allocate** (size_t __bytes, size_t __alignment) override
- void **do_deallocate** (void * __p, size_t __bytes, size_t __alignment) override
- bool **do_is_equal** (const [memory_resource](#) &__other) const noexcept override

6.1049.1 Detailed Description

A non-thread-safe memory resource that manages pools of fixed-size blocks.

6.1049.2 Member Function Documentation

do_allocate()

```
void * std::pmr::unsynchronized_pool_resource::do_allocate (
    size_t __bytes,
    size_t __alignment ) [override], [protected], [virtual]
```

Implements [std::pmr::memory_resource](#).

do_deallocate()

```
void std::pmr::unsynchronized_pool_resource::do_deallocate (
    void * __p,
    size_t __bytes,
    size_t __alignment ) [override], [protected], [virtual]
```

Implements [std::pmr::memory_resource](#).

do_is_equal()

```
bool std::pmr::unsynchronized_pool_resource::do_is_equal (
    const memory\_resource & __other ) const [inline], [override], [protected], [virtual],
[noexcept]
```

Implements [std::pmr::memory_resource](#).

The documentation for this class was generated from the following file:

- [memory_resource](#)

6.1050 std::unwrap_ref_decay< _Tp > Struct Template Reference**Public Types**

- using **type** = `unwrap_reference_t< decay_t< _Tp > >`

6.1050.1 Detailed Description

```
template<typename _Tp>
struct std::unwrap_ref_decay< _Tp >
```

Decay type and if it's a reference_wrapper, unwrap it

Since

C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.1051 std::unwrap_reference< _Tp > Struct Template Reference**Public Types**

- using **type** = `_Tp`

6.1051.1 Detailed Description

```
template<typename _Tp>
struct std::unwrap_reference< _Tp >
```

Unwrap a reference_wrapper

Since

C++20

The documentation for this struct was generated from the following file:

- [type_traits](#)

6.1052 `std::uses_allocator< typename, typename >` Struct Template Reference**6.1052.1 Detailed Description**

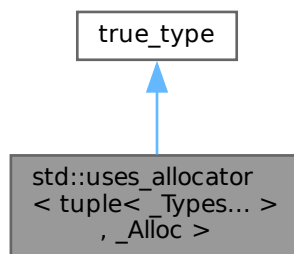
```
template<typename, typename>
struct std::uses_allocator< typename, typename >
```

Declare `uses_allocator` so it can be specialized in `<queue>` etc.
 The documentation for this struct was generated from the following file:

- [memoryfwd.h](#)

6.1053 `std::uses_allocator< tuple< _Types... >, _Alloc >` Struct Template Reference

Inheritance diagram for `std::uses_allocator< tuple< _Types... >, _Alloc >`:

**Public Types**

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const noexcept
- constexpr value_type **operator()** () const noexcept

Static Public Attributes

- static constexpr `_Tp` **value**

6.1053.1 Detailed Description

```
template<typename... _Types, typename _Alloc>
struct std::uses_allocator< tuple< _Types... >, _Alloc >
```

Partial specialization for tuples.
 The documentation for this struct was generated from the following file:

- [tuple](#)

6.1054 std::valarray<_Tp> Class Template Reference

Public Types

- typedef `_Tp` `value_type`

Public Member Functions

- `valarray` () noexcept
- template<class `_Dom` >
 `valarray` (const `_Expr`< `_Dom`, `_Tp` > &`__e`)
- `valarray` (const `_Tp` &, `size_t`)
- template<typename `_Tp` >
 `valarray` (const `_Tp` *`__restrict` `__p`, `size_t` `__n`)
- `valarray` (const `_Tp` *`__restrict` `__p`, `size_t`)
- `valarray` (const `gslice_array`< `_Tp` > &)
- `valarray` (const `indirect_array`< `_Tp` > &)
- `valarray` (const `mask_array`< `_Tp` > &)
- `valarray` (const `slice_array`< `_Tp` > &)
- `valarray` (const `valarray` &)
- `valarray` (`initializer_list`< `_Tp` >)
- `valarray` (`size_t`)
- `valarray` (`valarray` &&) noexcept
- `_Expr`< `_ValFunClos`< `_ValArray`, `_Tp` >, `_Tp` > `apply` (`_Tp` `__func`(`_Tp`)) const
- `_Expr`< `_RefFunClos`< `_ValArray`, `_Tp` >, `_Tp` > `apply` (`_Tp` `__func`(const `_Tp` &)) const
- `valarray`< `_Tp` > `cshift` (int `__n`) const
- `_Tp` `max` () const
- `_Tp` `min` () const
- `_UnaryOp`< `__logical_not` >::`Rt` `operator!` () const
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator%=(const _Expr`< `_Dom`, `_Tp` > &)
- `valarray`< `_Tp` > & `operator%=(const _Tp` &)
- `valarray`< `_Tp` > & `operator%=(const valarray`< `_Tp` > &)
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator&=(const _Expr`< `_Dom`, `_Tp` > &)
- `valarray`< `_Tp` > & `operator&=(const _Tp` &)
- `valarray`< `_Tp` > & `operator&=(const valarray`< `_Tp` > &)
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator*=(const _Expr`< `_Dom`, `_Tp` > &)
- `valarray`< `_Tp` > & `operator*=(const _Tp` &)
- `valarray`< `_Tp` > & `operator*=(const valarray`< `_Tp` > &)
- `_UnaryOp`< `__unary_plus` >::`Rt` `operator+` () const
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator+=(const _Expr`< `_Dom`, `_Tp` > &)
- `valarray`< `_Tp` > & `operator+=(const _Tp` &)
- `valarray`< `_Tp` > & `operator+=(const valarray`< `_Tp` > &)
- `_UnaryOp`< `__negate` >::`Rt` `operator-` () const
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator-=(const _Expr`< `_Dom`, `_Tp` > &)
- `valarray`< `_Tp` > & `operator-=(const _Tp` &)
- `valarray`< `_Tp` > & `operator-=(const valarray`< `_Tp` > &)
- template<class `_Dom` >
 `valarray`< `_Tp` > & `operator/=(const _Expr`< `_Dom`, `_Tp` > &)

- `valarray<_Tp> & operator/=(const _Tp &)`
- `valarray<_Tp> & operator/=(const valarray<_Tp> &)`
- `template<class _Dom>`
`valarray<_Tp> & operator<<=(const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator<<=(const _Tp &)`
- `valarray<_Tp> & operator<<=(const valarray<_Tp> &)`
- `template<class _Dom>`
`valarray<_Tp> & operator=(const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator=(const _Tp &__t)`
- `valarray<_Tp> & operator=(const gslice_array<_Tp> &__ga)`
- `valarray<_Tp> & operator=(const indirect_array<_Tp> &__ia)`
- `valarray<_Tp> & operator=(const mask_array<_Tp> &__ma)`
- `valarray<_Tp> & operator=(const slice_array<_Tp> &__sa)`
- `valarray<_Tp> & operator=(const valarray<_Tp> &__v)`
- `valarray & operator=(initializer_list<_Tp> __l)`
- `valarray<_Tp> & operator=(valarray<_Tp> &&__v) noexcept`
- `template<class _Dom>`
`valarray<_Tp> & operator>>=(const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator>>=(const _Tp &)`
- `valarray<_Tp> & operator>>=(const valarray<_Tp> &)`
- `gslice_array<_Tp> operator[] (const gslice &__s)`
- `_Expr<_GClos<_ValArray, _Tp>, _Tp> operator[] (const gslice &__s) const`
- `mask_array<_Tp> operator[] (const valarray<bool> &__m)`
- `valarray<_Tp> operator[] (const valarray<bool> &__m) const`
- `indirect_array<_Tp> operator[] (const valarray<size_t> &__i)`
- `_Expr<_IClos<_ValArray, _Tp>, _Tp> operator[] (const valarray<size_t> &__i) const`
- `_Tp & operator[] (size_t __i) noexcept`
- `const _Tp & operator[] (size_t) const noexcept`
- `slice_array<_Tp> operator[] (slice __s)`
- `_Expr<_SClos<_ValArray, _Tp>, _Tp> operator[] (slice __s) const`
- `template<class _Dom>`
`valarray<_Tp> & operator^=(const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator^=(const _Tp &)`
- `valarray<_Tp> & operator^=(const valarray<_Tp> &)`
- `template<class _Dom>`
`valarray<_Tp> & operator|= (const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator|= (const _Tp &)`
- `valarray<_Tp> & operator|= (const valarray<_Tp> &)`
- `_UnaryOp<__bitwise_not>::_Rt operator~ () const`
- `void resize (size_t __size, _Tp __c=_Tp())`
- `valarray<_Tp> shift (int __n) const`
- `size_t size () const`
- `_Tp sum () const`
- `void swap (valarray<_Tp> &__v) noexcept`

Friends

- `struct _Array<_Tp>`

6.1054.1 Detailed Description

```
template<class _Tp>
class std::valarray<_Tp>
```

Smart array designed to support numeric processing.

A valarray is an array that provides constraints intended to allow for effective optimization of numeric array processing by reducing the aliasing that can result from pointer representations. It represents a one-dimensional array from which different multidimensional subsets can be accessed and modified.

Template Parameters

<code>_Tp</code>	Type of object in the array.
------------------	------------------------------

6.1054.2 Constructor & Destructor Documentation

`valarray()`

```
template<class _Tp>
std::valarray<_Tp>::valarray (
    const _Tp * __restrict__,
    size_t )
```

Construct an array initialized to the first n elements of t .

6.1054.3 Member Function Documentation

`operator"!")()`

```
template<class _Tp>
_UnaryOp< __logical_not >::_Rt std::valarray<_Tp>::operator! ( ) const
```

Return a new valarray by applying unary ! to each element.

`operator%=()` [1/2]

```
template<class _Tp>
valarray<_Tp> & std::valarray<_Tp>::operator%= (
    const _Tp & )
```

Set each element e of array to $e \% t$.

`operator%=()` [2/2]

```
template<class _Tp>
valarray<_Tp> & std::valarray<_Tp>::operator%= (
    const valarray<_Tp> & )
```

Modulo elements of array by corresponding elements of v .

`operator&=()` [1/2]

```
template<class _Tp>
valarray<_Tp> & std::valarray<_Tp>::operator&= (
    const _Tp & )
```

Set each element e of array to $e \& t$.

operator&=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator&= (
    const valarray< _Tp > & )
```

Logical and corresponding elements of *v* with elements of array.

operator*=() [1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator*= (
    const _Tp & )
```

Multiply each element of array by *t*.

operator*=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator*= (
    const valarray< _Tp > & )
```

Multiply elements of array by corresponding elements of *v*.

operator+()

```
template<class _Tp >
UnaryOp< __unary_plus >::_Rt std::valarray< _Tp >::operator+ ( ) const
```

Return a new valarray by applying unary + to each element.

operator+=() [1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator+= (
    const _Tp & )
```

Add *t* to each element of array.

operator+=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator+= (
    const valarray< _Tp > & )
```

Add corresponding elements of *v* to elements of array.

operator-()

```
template<class _Tp >
UnaryOp< __negate >::_Rt std::valarray< _Tp >::operator- ( ) const
```

Return a new valarray by applying unary - to each element.

operator-=() [1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator-= (
    const _Tp & )
```

Subtract *t* to each element of array.

operator-=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator-= (
    const valarray< _Tp > & )
```

Subtract corresponding elements of *v* from elements of array.

operator/=([1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator/= (
    const _Tp & )
```

Divide each element of array by *t*.

operator/=([2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator/= (
    const valarray< _Tp > & )
```

Divide elements of array by corresponding elements of *v*.

operator<<=([1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator<<= (
    const _Tp & )
```

Left shift each element *e* of array by *t* bits.

operator<<=([2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator<<= (
    const valarray< _Tp > & )
```

Left shift elements of array by corresponding elements of *v*.

operator>>=([1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator>>= (
    const _Tp & )
```

Right shift each element *e* of array by *t* bits.

operator>>=([2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator>>= (
    const valarray< _Tp > & )
```

Right shift elements of array by corresponding elements of *v*.

operator^=([1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator^= (
    const _Tp & )
```

Set each element *e* of array to $e \wedge t$.

operator^=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator^= (
    const valarray< _Tp > & )
```

Logical xor corresponding elements of *v* with elements of array.

operator" |=() [1/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator|= (
    const _Tp & )
```

Set each element *e* of array to *e* | *t*.

operator" |=() [2/2]

```
template<class _Tp >
valarray< _Tp > & std::valarray< _Tp >::operator|= (
    const valarray< _Tp > & )
```

Logical or corresponding elements of *v* with elements of array.

operator~()

```
template<class _Tp >
__UnaryOp< __bitwise_not >::Rt std::valarray< _Tp >::operator~ ( ) const
```

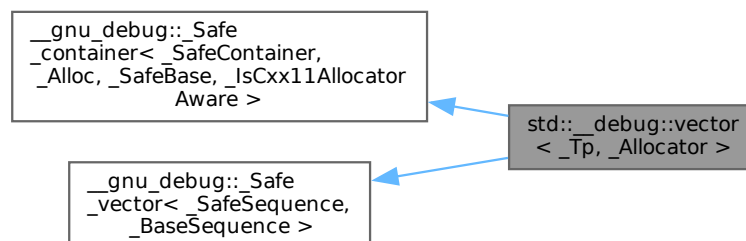
Return a new valarray by applying unary ~ to each element.

The documentation for this class was generated from the following file:

- [valarray](#)

6.1055 std::__debug::vector< _Tp, _Allocator > Class Template Reference

Inheritance diagram for std::__debug::vector< _Tp, _Allocator >:

**Public Types**

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug:: Safe_iterator< _Base_const_iterator, vector >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**

- typedef `std::reverse_iterator< const_iterator >` `const_reverse_iterator`
- typedef `_Base::difference_type` `difference_type`
- typedef `__gnu_debug::_Safe_iterator< _Base_iterator, vector >` `iterator`
- typedef `_Base::pointer` `pointer`
- typedef `_Base::reference` `reference`
- typedef `std::reverse_iterator< iterator >` `reverse_iterator`
- typedef `_Base::size_type` `size_type`
- typedef `_Tp` `value_type`

Public Member Functions

- `vector` (`_Base_ref __x`)
- template<class `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>`>
 `vector` (`_InputIterator __first`, `_InputIterator __last`, const `_Allocator` & `__a`=`_Allocator()`)
- `vector` (const `_Allocator` & `__a`) noexcept
- `vector` (const `vector` &)=default
- `vector` (const `vector` & `__x`, const `__type_identity_t< allocator_type >` & `__a`)
- `vector` (`initializer_list< value_type >` `__l`, const `allocator_type` & `__a`=`allocator_type()`)
- `vector` (`size_type __n`, const `__type_identity_t< _Tp >` & `__value`, const `_Allocator` & `__a`=`_Allocator()`)
- `vector` (`size_type __n`, const `_Allocator` & `__a`=`_Allocator()`)
- `vector` (`vector` &&)=default
- `vector` (`vector` && `__x`, const `__type_identity_t< allocator_type >` & `__a`) noexcept(`std::is_nothrow_constructible< _Base, _Base, const allocator_type >::value`)
- const `_Base` & `_M_base` () const noexcept
- `_Base` & `_M_base` () noexcept
- template<typename `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>`>>
 void `assign` (`_InputIterator __first`, `_InputIterator __last`)
- void `assign` (`initializer_list< value_type >` `__l`)
- void `assign` (`size_type __n`, const `_Tp` & `__u`)
- const_reference `back` () const noexcept
- reference `back` () noexcept
- `const_iterator` `begin` () const noexcept
- `iterator` `begin` () noexcept
- `size_type` `capacity` () const noexcept
- `const_iterator` `cbegin` () const noexcept
- `const_iterator` `cend` () const noexcept
- void `clear` () noexcept
- `const_reverse_iterator` `crbegin` () const noexcept
- `const_reverse_iterator` `crend` () const noexcept
- template<typename... `_Args`>
 `iterator` `emplace` (`const_iterator` `__position`, `_Args` &&... `__args`)
- template<typename... `_Args`>
 reference `emplace_back` (`_Args` &&... `__args`)
- `const_iterator` `end` () const noexcept
- `iterator` `end` () noexcept
- `iterator` `erase` (`const_iterator` `__first`, `const_iterator` `__last`)
- `iterator` `erase` (`const_iterator` `__position`)
- const_reference `front` () const noexcept
- reference `front` () noexcept
- template<class `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>`>>
 `iterator` `insert` (`const_iterator` `__position`, `_InputIterator` `__first`, `_InputIterator` `__last`)

- `template<typename _Up = _Tp>`
`__gnu_cxx::__enable_if<!std::__are_same<_Up, bool>::__value, iterator>::__type insert (const_iterator __`
`__position, _Tp &&__x)`
- `iterator insert (const_iterator __position, const _Tp &__x)`
- `iterator insert (const_iterator __position, initializer_list<value_type> __l)`
- `iterator insert (const_iterator __position, size_type __n, const _Tp &__x)`
- `vector & operator= (const vector &)=default`
- `vector & operator= (initializer_list<value_type> __l)`
- `vector & operator= (vector &&)=default`
- `const_reference operator[] (size_type __n) const noexcept`
- `reference operator[] (size_type __n) noexcept`
- `void pop_back () noexcept`
- `template<typename _Up = _Tp>`
`__gnu_cxx::__enable_if<!std::__are_same<_Up, bool>::__value, void>::__type push_back (_Tp &&__x)`
- `void push_back (const _Tp &__x)`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `reverse_iterator rend () noexcept`
- `void reserve (size_type __n)`
- `void resize (size_type __sz)`
- `void resize (size_type __sz, const _Tp &__c)`
- `void shrink_to_fit ()`
- `void swap (vector &__x) noexcept(*conditional *)`

Protected Member Functions

- `bool _M_requires_reallocation (size_type __elements) const noexcept`
- `void _M_swap (_Safe_container &__x) noexcept`
- `void _M_update_guaranteed_capacity () noexcept`

Protected Attributes

- `size_type _M_guaranteed_capacity`

Friends

- `template<typename _ItT, typename _SeqT, typename _CatT>`
`class ::__gnu_debug::_Safe_iterator`

6.1055.1 Detailed Description

`template<typename _Tp, typename _Allocator = std::allocator<_Tp>>`
`class std::__debug::vector<_Tp, _Allocator>`

Class `std::vector` with safety/checking/debug instrumentation.

6.1055.2 Constructor & Destructor Documentation

vector()

```
template<typename _Tp , typename _Allocator = std::allocator<_Tp>>
std::__debug::vector< _Tp, _Allocator >::vector (
    _Base_ref __x ) [inline]
```

Construction from a normal-mode vector.

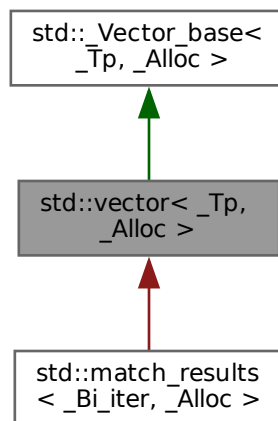
The documentation for this class was generated from the following file:

- [debug/vector](#)

6.1056 std::vector< _Tp, _Alloc > Class Template Reference

```
#include <stl_vector.h>
```

Inheritance diagram for std::vector< _Tp, _Alloc >:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `__gnu_cxx::__normal_iterator< const_pointer, vector >` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `__gnu_cxx::__normal_iterator< pointer, vector >` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- [vector](#) ()=default
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
constexpr [vector](#) (_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type())
- constexpr [vector](#) (const allocator_type &__a) noexcept
- constexpr [vector](#) (const [vector](#) &__x)
- constexpr [vector](#) (const [vector](#) &__x, const __type_identity_t< allocator_type > &__a)
- constexpr [vector](#) (initializer_list< value_type > __l, const allocator_type &__a=allocator_type())
- constexpr [vector](#) (size_type __n, const allocator_type &__a=allocator_type())
- constexpr [vector](#) (size_type __n, const value_type &__value, const allocator_type &__a=allocator_type())
- [vector](#) ([vector](#) &&) noexcept=default
- constexpr [vector](#) ([vector](#) &&__rv, const __type_identity_t< allocator_type > &__m) noexcept(noexcept([vector](#)(std::declval< [vector](#) && >(), std::declval< const allocator_type & >(), std::declval< typename _Alloc_traits::is_always_equal >())))
- constexpr ~[vector](#) () noexcept
- template<typename... _Args>
constexpr auto [M_emplace_aux](#) (const_iterator __position, _Args &&... __args) -> iterator
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
constexpr void [assign](#) (_InputIterator __first, _InputIterator __last)
- constexpr void [assign](#) (initializer_list< value_type > __l)
- constexpr void [assign](#) (size_type __n, const value_type &__val)
- constexpr reference [at](#) (size_type __n)
- constexpr const_reference [at](#) (size_type __n) const
- constexpr const_reference [back](#) () const noexcept
- constexpr reference [back](#) () noexcept
- constexpr const_iterator [begin](#) () const noexcept
- constexpr iterator [begin](#) () noexcept
- constexpr size_type [capacity](#) () const noexcept
- constexpr const_iterator [cbegin](#) () const noexcept
- constexpr const_iterator [cend](#) () const noexcept
- constexpr void [clear](#) () noexcept
- constexpr const_reverse_iterator [crbegin](#) () const noexcept
- constexpr const_reverse_iterator [crend](#) () const noexcept
- constexpr const_Tp * [data](#) () const noexcept
- constexpr_Tp * [data](#) () noexcept
- template<typename... _Args>
constexpr iterator [emplace](#) (const_iterator __position, _Args &&... __args)
- template<typename... _Args>
constexpr reference [emplace_back](#) (_Args &&... __args)
- constexpr bool [empty](#) () const noexcept
- constexpr const_iterator [end](#) () const noexcept
- constexpr iterator [end](#) () noexcept
- constexpr iterator [erase](#) (const_iterator __first, const_iterator __last)
- constexpr iterator [erase](#) (const_iterator __position)
- constexpr const_reference [front](#) () const noexcept
- constexpr reference [front](#) () noexcept
- constexpr allocator_type [get_allocator](#) () const noexcept
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
constexpr iterator [insert](#) (const_iterator __position, _InputIterator __first, _InputIterator __last)
- constexpr iterator [insert](#) (const_iterator __position, const value_type &__x)
- constexpr iterator [insert](#) (const_iterator __position, initializer_list< value_type > __l)

- constexpr iterator `insert` (const_iterator __position, size_type __n, const value_type &__x)
- constexpr iterator `insert` (const_iterator __position, value_type &&__x)
- constexpr size_type `max_size` () const noexcept
- constexpr `vector` & `operator=` (const `vector` &__x)
- constexpr `vector` & `operator=` (initializer_list< value_type > __l)
- constexpr `vector` & `operator=` (`vector` &&__x) noexcept(_Alloc_traits::_S_nothrow_move())
- constexpr const_reference `operator[]` (size_type __n) const noexcept
- constexpr reference `operator[]` (size_type __n) noexcept
- constexpr void `pop_back` () noexcept
- constexpr void `push_back` (const value_type &__x)
- constexpr void `push_back` (value_type &&__x)
- constexpr `const_reverse_iterator` `rbegin` () const noexcept
- constexpr `reverse_iterator` `rbegin` () noexcept
- constexpr `const_reverse_iterator` `rend` () const noexcept
- constexpr `reverse_iterator` `rend` () noexcept
- constexpr void `reserve` (size_type __n)
- constexpr void `resize` (size_type __new_size)
- constexpr void `resize` (size_type __new_size, const value_type &__x)
- constexpr void `shrink_to_fit` ()
- constexpr size_type `size` () const noexcept
- constexpr void `swap` (`vector` &__x) noexcept

Protected Member Functions

- constexpr pointer `_M_allocate` (size_t __n)
- template<typename _ForwardIterator >
constexpr pointer `_M_allocate_and_copy` (size_type __n, _ForwardIterator __first, _ForwardIterator __last)
- template<typename _ForwardIterator >
constexpr void `_M_assign_aux` (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)
- template<typename _InputIterator >
constexpr void `_M_assign_aux` (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)
- template<typename _InputIterator >
constexpr void `_M_assign_dispatch` (_InputIterator __first, _InputIterator __last, __false_type)
- template<typename _Integer >
constexpr void `_M_assign_dispatch` (_Integer __n, _Integer __val, __true_type)
- constexpr size_type `_M_check_len` (size_type __n, const char *__s) const
- constexpr void `_M_create_storage` (size_t __n)
- constexpr void `_M_deallocate` (pointer __p, size_t __n)
- constexpr void `_M_default_append` (size_type __n)
- constexpr void `_M_default_initialize` (size_type __n)
- template<typename... _Args>
constexpr iterator `_M_emplace_aux` (const_iterator __position, _Args &&... __args)
- constexpr iterator `_M_emplace_aux` (const_iterator __position, value_type &&__v)
- constexpr iterator `_M_erase` (iterator __first, iterator __last)
- constexpr iterator `_M_erase` (iterator __position)
- constexpr void `_M_erase_at_end` (pointer __pos) noexcept
- constexpr void `_M_fill_assign` (size_type __n, const value_type &__val)
- constexpr void `_M_fill_initialize` (size_type __n, const value_type &__value)
- constexpr void `_M_fill_insert` (iterator __pos, size_type __n, const value_type &__x)
- constexpr const_Tp_alloc_type & `_M_get_Tp_allocator` () const noexcept
- constexpr_Tp_alloc_type & `_M_get_Tp_allocator` () noexcept

- `template<typename _Arg >`
`constexpr void _M_insert_aux (iterator __position, _Arg && __arg)`
- `template<typename _InputIterator >`
`constexpr void _M_insert_dispatch (iterator __pos, _InputIterator __first, _InputIterator __last, __false_type)`
- `template<typename _Integer >`
`constexpr void _M_insert_dispatch (iterator __pos, _Integer __n, _Integer __val, __true_type)`
- `constexpr iterator _M_insert_rval (const_iterator __position, value_type && __v)`
- `constexpr void _M_range_check (size_type __n) const`
- `template<typename _ForwardIterator >`
`constexpr void _M_range_initialize (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr void _M_range_initialize (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`constexpr void _M_range_insert (iterator __pos, _ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr void _M_range_insert (iterator __pos, _InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename... _Args>`
`constexpr void _M_realloc_insert (iterator __position, _Args &&... __args)`
- `constexpr bool _M_shrink_to_fit ()`

Static Protected Member Functions

- `static constexpr size_type _S_check_init_len (size_type __n, const allocator_type & __a)`
- `static constexpr size_type _S_max_size (const _Tp_alloc_type & __a) noexcept`

Protected Attributes

- `_Vector_impl _M_impl`

6.1056.1 Detailed Description

`template<typename _Tp, typename _Alloc = std::allocator<_Tp>>`
class `std::vector< _Tp, _Alloc >`

A standard container which offers fixed time access to individual elements in any order.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#) with the exception of `push_front` and `pop_front`.

In some terminology a vector can be described as a dynamic C-style array, it offers fast and efficient access to individual elements in any order and saves the user from worrying about memory and size allocation. Subscripting (`[]`) access is also provided as with C-style arrays.

6.1056.2 Constructor & Destructor Documentation

vector() [1/10]

`template<typename _Tp , typename _Alloc = std::allocator<_Tp>>`
[std::vector](#)< _Tp, _Alloc >::vector () [default]

Creates a vector with no elements.

vector() [2/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector< _Tp, _Alloc >::vector (
    const allocator_type & __a ) [inline], [explicit], [constexpr], [noexcept]
```

Creates a vector with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

vector() [3/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector< _Tp, _Alloc >::vector (
    size_type __n,
    const allocator_type & __a = allocator_type() ) [inline], [explicit], [constexpr]
```

Creates a vector with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__a</code>	An allocator.

This constructor fills the vector with `__n` default constructed elements.

vector() [4/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector< _Tp, _Alloc >::vector (
    size_type __n,
    const value_type & __value,
    const allocator_type & __a = allocator_type() ) [inline], [constexpr]
```

Creates a vector with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__a</code>	An allocator.

This constructor fills the vector with `__n` copies of `__value`.

vector() [5/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector< _Tp, _Alloc >::vector (
    const vector< _Tp, _Alloc > & __x ) [inline], [constexpr]
```

Vector copy constructor.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but any unused capacity in `__x` will not be copied (i.e. `capacity() == size()` in the new vector).

The newly-created vector uses a copy of the allocator object used by `__x` (unless the allocator traits dictate a different object).

References [std::vector<_Tp, _Alloc >::begin\(\)](#), and [std::vector<_Tp, _Alloc >::end\(\)](#).

vector() [6/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
std::vector<_Tp, _Alloc >::vector (
    vector<_Tp, _Alloc > && ) [default], [noexcept]
```

Vector move constructor.

The newly-created vector contains the exact contents of the moved instance. The contents of the moved instance are a valid, but unspecified vector.

vector() [7/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector<_Tp, _Alloc >::vector (
    const vector<_Tp, _Alloc > & __x,
    const __type_identity_t< allocator_type > & __a ) [inline], [constexpr]
```

Copy constructor with alternative allocator.

vector() [8/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector<_Tp, _Alloc >::vector (
    vector<_Tp, _Alloc > && __rv,
    const __type_identity_t< allocator_type > & __m ) [inline], [constexpr], [noexcept]
```

Move constructor with alternative allocator.

vector() [9/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector<_Tp, _Alloc >::vector (
    initializer_list< value_type > __l,
    const allocator_type & __a = allocator_type() ) [inline], [constexpr]
```

Builds a vector from an initializer list.

Parameters

<code>__l</code>	An initializer_list.
<code>__a</code>	An allocator.

Create a vector consisting of copies of the elements in the initializer_list `__l`.

This will call the element type's copy constructor N times (where N is `__l.size()`) and do no memory reallocation.

vector() [10/10]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
constexpr std::vector< _Tp, _Alloc >::vector (
    _InputIterator __first,
    _InputIterator __last,
    const allocator_type & __a = allocator_type() ) [inline], [constexpr]
```

Builds a vector from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__a</code>	An allocator.

Create a vector consisting of copies of the elements from [first,last).

If the iterators are forward, bidirectional, or random-access, then this will call the elements' copy constructor *N* times (where *N* is distance(first,last)) and do no memory reallocation. But if only input iterators are used, then this will do at most 2*N* calls to the copy constructor, and log*N* memory reallocations.

References [std::_iterator_category\(\)](#).

~vector()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr std::vector< _Tp, _Alloc >::~~vector ( ) [inline], [constexpr], [noexcept]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::_Destroy\(\)](#).

6.1056.3 Member Function Documentation**_M_allocate_and_copy()**

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _ForwardIterator >
constexpr pointer std::vector< _Tp, _Alloc >::_M_allocate_and_copy (
    size_type __n,
    _ForwardIterator __first,
    _ForwardIterator __last ) [inline], [constexpr], [protected]
```

Memory expansion handler. Uses the member allocation function to obtain *n* bytes of memory, and then copies [first,last) into it.

_M_range_check()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::_M_range_check (
    size_type __n ) const [inline], [constexpr], [protected]
```

Safety check used only from at().

References [std::vector< _Tp, _Alloc >::size\(\)](#).

Referenced by [std::vector< _Tp, _Alloc >::at\(\)](#), and [std::vector< _Tp, _Alloc >::at\(\)](#).

assign() [1/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
```

```
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>>
constexpr void std::vector< _Tp, _Alloc >::assign (
    _InputIterator __first,
    _InputIterator __last ) [inline], [constexpr]
```

Assigns a range to a vector.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a vector with copies of the elements in the range `[__first,__last)`.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned.

assign() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>>
constexpr void std::vector< _Tp, _Alloc >::assign (
    initializer_list< value_type > __l ) [inline], [constexpr]
```

Assigns an initializer list to a vector.

Parameters

<code>↵</code>	An initializer_list.
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code>/</code>	

This function fills a vector with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned.

assign() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>>
constexpr void std::vector< _Tp, _Alloc >::assign (
    size_type __n,
    const value_type & __val ) [inline], [constexpr]
```

Assigns a given value to a vector.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a vector with `__n` copies of the given value. Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned.


at() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>>
```

```
constexpr reference std::vector< _Tp, _Alloc >::at (
    size_type __n ) [inline], [constexpr]
```

Provides access to the data contained in the vector.

Parameters

 __n	The index of the element for which data should be accessed.
--	---

Returns

Read/write reference to data.

Exceptions

std::out_of_range	If __n is an invalid index.
-----------------------------------	---

This function provides for safer data access. The parameter is first checked that it is in the range of the vector. The function throws [out_of_range](#) if the check fails.


References [std::vector< _Tp, _Alloc >::_M_range_check\(\)](#).

at() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reference std::vector< _Tp, _Alloc >::at (
    size_type __n ) const [inline], [constexpr]
```

Provides access to the data contained in the vector.

Parameters

 __n	The index of the element for which data should be accessed.
--	---

Returns

Read-only (constant) reference to data.

Exceptions

std::out_of_range	If __n is an invalid index.
-----------------------------------	---

This function provides for safer data access. The parameter is first checked that it is in the range of the vector. The function throws [out_of_range](#) if the check fails.

References [std::vector< _Tp, _Alloc >::_M_range_check\(\)](#).

back() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reference std::vector< _Tp, _Alloc >::back ( ) const [inline], [constexpr],
[noexcept]
```

Returns a read-only (constant) reference to the data at the last element of the vector.

References [std::vector< _Tp, _Alloc >::end\(\)](#).

back() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr reference std::vector< _Tp, _Alloc >::back ( ) [inline], [constexpr], [noexcept]
```

Returns a read/write reference to the data at the last element of the vector.

References `std::vector<_Tp, _Alloc>::end()`.

Referenced by `std::piecewise_constant_distribution<_RealType>::max()`, and `std::piecewise_linear_distribution<_RealType>::max()`.

```
begin() [1/2]
```

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_iterator std::vector< _Tp, _Alloc >::begin ( ) const [inline], [constexpr],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the vector. Iteration is done in ordinary element order.

begin() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector< _Tp, _Alloc >::begin ( ) [inline], [constexpr], [noexcept]
```

Returns a read/write iterator that points to the first element in the vector. Iteration is done in ordinary element order.

Referenced by `std::vector<_Tp,_Alloc>::vector()`, `std::match_results<_Bi_iter,_Alloc>::begin()`, `std::vector<_Tp,_Alloc>::crend()`, `std::vector<_Tp,_Alloc>::empty()`, `std::vector<_Tp,_Alloc>::erase()`, `std::vector<_Tp,_Alloc>::erase()`, `std::vector<_Tp,_Alloc>::front()`, `std::vector<_Tp,_Alloc>::insert()`, `std::vector<_Tp,_Alloc>::insert()`, `std::vector<_Tp,_Alloc>::__gnu_parallel::multiway_merge_exact_splitting()`, `std::operator<()`, `std::vector<_Tp,_Alloc>::operator=()`, `std::operator==(())`, `std::vector<_Tp,_Alloc>::rend()`, and `std::vector<_Tp,_Alloc>::rend()`.

capacity()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr size_type std::vector< _Tp, _Alloc >::capacity ( ) const [inline], [constexpr], [noexcept]
```

Returns the total number of elements that the vector can hold before needing to allocate more memory.

cbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_iterator std::vector<_Tp, _Alloc >::cbegin ( ) const [inline], [constexpr],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the vector. Iteration is done in ordinary element order.

Referenced by `std::vector<_Tp, _Alloc>::erase()`, `std::vector<_Tp, _Alloc>::erase()`, `std::vector<_Tp, _Alloc>::insert()`, `std::vector<_Tp, _Alloc>::insert()`, and `std::vector<_Tp, _Alloc>::insert()`.

cend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_iterator std::vector< _Tp, _Alloc >::cend ( ) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

clear()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::clear ( ) [inline], [constexpr], [noexcept]
```

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

crbegin()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reverse_iterator std::vector< _Tp, _Alloc >::crbegin ( ) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

References [std::vector< _Tp, _Alloc >::end\(\)](#).

crend()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reverse_iterator std::vector< _Tp, _Alloc >::crend ( ) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

References [std::vector< _Tp, _Alloc >::begin\(\)](#).

data()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr _Tp * std::vector< _Tp, _Alloc >::data ( ) [inline], [constexpr], [noexcept]
```

Returns a pointer such that `[data(), data() + size())` is a valid range. For a non-empty vector, `data() == &front()`.

emplace()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename... _Args>
constexpr iterator std::vector< _Tp, _Alloc >::emplace (
    const_iterator __position,
    _Args &&... __args ) [inline], [constexpr]
```

Inserts an object in vector before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type `T` constructed with `T(std::forward<Args>(args)...) before the specified location`. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

empty()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr bool std::vector< _Tp, _Alloc >::empty ( ) const [inline], [constexpr], [noexcept]
```

Returns true if the vector is empty. (Thus `begin()` would equal `end()`.)

References [std::vector< _Tp, _Alloc >::begin\(\)](#), and [std::vector< _Tp, _Alloc >::end\(\)](#).

Referenced by [std::piecewise_constant_distribution< _RealType >::densities\(\)](#), [std::piecewise_linear_distribution< _RealType >::densities\(\)](#), [std::match_results< _Bi_iter, _Alloc >::end\(\)](#), [std::piecewise_constant_distribution< _RealType >::intervals\(\)](#), [std::piecewise_linear_distribution< _RealType >::intervals\(\)](#), [std::discrete_distribution< _IntType >::max\(\)](#), [std::piecewise_constant_distribution< _RealType >::max\(\)](#), [std::piecewise_linear_distribution< _RealType >::max\(\)](#).

[std::piecewise_constant_distribution<_RealType>::min\(\)](#), [std::piecewise_linear_distribution<_RealType>::min\(\)](#), [std::discrete_distribution<_IntType>::probabilities\(\)](#), [std::match_results<_Bi_iter, _Alloc>::ready\(\)](#), and [std::match_results<_Bi_iter,](#)

end() [1/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr const_iterator std::vector<\_Tp, \_Alloc>::end \( \) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

end() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<\_Tp, \_Alloc>::end \( \) [inline], [constexpr], [noexcept]
```

Returns a read/write iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

Referenced by [std::vector<_Tp, _Alloc>::vector\(\)](#), [std::vector<_Tp, _Alloc>::back\(\)](#), [std::vector<_Tp, _Alloc>::back\(\)](#), [std::vector<_Tp, _Alloc>::cbegin\(\)](#), [std::vector<_Tp, _Alloc>::empty\(\)](#), [std::match_results<_Bi_iter, _Alloc>::end\(\)](#), [__gnu_parallel::multiway_merge_exact_splitting\(\)](#), [std::operator<\(\)](#), [std::vector<_Tp, _Alloc>::operator=\(\)](#), [std::operator==\(\)](#), [std::vector<_Tp, _Alloc>::push_back\(\)](#), [std::vector<_Tp, _Alloc>::rbegin\(\)](#), [std::vector<_Tp, _Alloc>::rbegin\(\)](#), and [std::vector<_Tp, _Alloc>::resize\(\)](#).

erase() [1/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<\_Tp, \_Alloc>::erase \(
    const_iterator __first,
    const_iterator __last ) [inline], [constexpr]
```

Remove a range of elements.

Parameters

<i>__first</i>	Iterator pointing to the first element to be erased.
<i>__last</i>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by *__last* prior to erasing (or [end\(\)](#)).

This function will erase the elements in the range [*__first*,*__last*) and shorten the vector accordingly.

Note This operation could be expensive and if it is frequently used the user should consider using [std::list](#). The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References [std::vector<_Tp, _Alloc>::begin\(\)](#), and [std::vector<_Tp, _Alloc>::cbegin\(\)](#).

erase() [2/2]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<\_Tp, \_Alloc>::erase \(
    const_iterator __position ) [inline], [constexpr]
```

Remove element at given position.

Parameters

<i>__position</i>	Iterator pointing to element to be erased.
-------------------	--

Returns

An iterator pointing to the next element (or end()).

This function will erase the element at the given position and thus shorten the vector by one.

Note This operation could be expensive and if it is frequently used the user should consider using `std::list`. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

References `std::vector<_Tp, _Alloc>::begin()`, and `std::vector<_Tp, _Alloc>::cbegin()`.

front() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reference std::vector<_Tp, _Alloc>::front ( ) const [inline], [constexpr],
[noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the vector.

References `std::vector<_Tp, _Alloc>::begin()`.

front() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr reference std::vector<_Tp, _Alloc>::front ( ) [inline], [constexpr], [noexcept]
```

Returns a read/write reference to the data at the first element of the vector.

References `std::vector<_Tp, _Alloc>::begin()`.

Referenced by `std::piecewise_constant_distribution<_RealType>::min()`, and `std::piecewise_linear_distribution<_RealType>::min()`.

get_allocator()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr allocator_type std::_Vector_base<_Tp, _Alloc>::get_allocator ( ) const [inline],
[constexpr], [noexcept]
```

Get a copy of the memory allocation object.

Referenced by `std::match_results<_Bi_iter, _Alloc>::get_allocator()`.

insert() [1/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>
constexpr iterator std::vector<_Tp, _Alloc>::insert (
    const_iterator __position,
    _InputIterator __first,
    _InputIterator __last ) [inline], [constexpr]
```

Inserts a range into the vector.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator that points to the inserted data.

This function will insert copies of the data in the range `[__first,__last)` into the vector before the location specified by `pos`.

Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

References [std::vector<_Tp, _Alloc>::begin\(\)](#), and [std::vector<_Tp, _Alloc>::cbegin\(\)](#).

insert() [2/5]

```
template<typename _Tp, typename _Alloc>
constexpr vector<_Tp, _Alloc>::iterator vector::insert (
    const_iterator __position,
    const value_type & __x ) [constexpr]
```

Inserts given value into vector before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

References [std::begin\(\)](#), [std::cbegin\(\)](#), [std::end\(\)](#), and [std::move\(\)](#).

insert() [3/5]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<_Tp, _Alloc>::insert (
    const_iterator __position,
    initializer\_list<value_type> __l ) [inline], [constexpr]
```

Inserts an `initializer_list` into the vector.

Parameters

<code>__position</code>	An iterator into the vector.
<code>__l</code>	An <code>initializer_list</code> .

This function will insert copies of the data in the `initializer_list l` into the vector before the location specified by `position`. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

References [std::vector<_Tp, _Alloc>::begin\(\)](#), and [std::vector<_Tp, _Alloc>::cbegin\(\)](#).

insert() [4/5]

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<_Tp, _Alloc>::insert (
    const_iterator __position,
    size_type __n,
    const value_type & __x ) [inline], [constexpr]
```

Inserts a number of copies of given data into the vector.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
-------------------------	--

Parameters

<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a specified number of copies of the given data before the location specified by *position*.

Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

References [std::vector<_Tp, _Alloc>::begin\(\)](#), and [std::vector<_Tp, _Alloc>::cbegin\(\)](#).

insert() [5/5]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr iterator std::vector<\_Tp, \_Alloc>::insert (
    const_iterator __position,
    value_type && __x ) [inline], [constexpr]
```

Inserts given rvalue into vector before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

References [std::move\(\)](#).

max_size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr size_type std::vector<\_Tp, \_Alloc>::max\_size ( ) const [inline], [constexpr], [noexcept]
```

Returns the `size()` of the largest possible vector.

Referenced by [std::match_results<_Bi_iter, _Alloc>::max_size\(\)](#).

operator=() [1/3]

```
template<typename _Tp , typename _Alloc >
constexpr vector<\_Tp, \_Alloc> & vector::operator= (
    const vector<\_Tp, \_Alloc> & __x ) [constexpr]
```

Vector assignment operator.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but any unused capacity in `__x` will not be copied.

Whether the allocator is copied depends on the allocator traits.

References [std::__addressof\(\)](#), [std::_Destroy\(\)](#), [std::vector<_Tp, _Alloc >::begin\(\)](#), [std::begin\(\)](#), [std::vector<_Tp, _Alloc >::end\(\)](#), [std::end\(\)](#), [std::vector<_Tp, _Alloc >::size\(\)](#), and [std::size\(\)](#).

operator=() [2/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr vector & std::vector<_Tp, _Alloc >::operator= (
    initializer_list< value_type > __l ) [inline], [constexpr]
```

Vector list assignment operator.

Parameters

<code>↵</code>	An initializer_list.
<code>↵</code>	
<code>↵</code>	
<code>↵</code>	
<code>l</code>	

This function fills a vector with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned.

operator=() [3/3]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr vector & std::vector<_Tp, _Alloc >::operator= (
    vector<_Tp, _Alloc > && __x ) [inline], [constexpr], [noexcept]
```

Vector move assignment operator.

Parameters

<code>↵</code>	A vector of identical element and allocator types.
<code>x</code>	

The contents of `__x` are moved into this vector (without copying, if the allocators permit it). Afterwards `__x` is a valid, but unspecified vector.

Whether the allocator is moved depends on the allocator traits.

References [std::move\(\)](#).

operator[]() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reference std::vector<_Tp, _Alloc >::operator[] (
    size_type __n ) const [inline], [constexpr], [noexcept]
```

Subscript access to the data contained in the vector.

Parameters

<code>↵</code>	The index of the element for which data should be accessed.
<code>n</code>	

Returns

Read-only (constant) reference to data.


This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

operator[]() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr reference std::vector< _Tp, _Alloc >::operator[] (
    size_type __n ) [inline], [constexpr], [noexcept]
```

Subscript access to the data contained in the vector.

Parameters

 <code>__n</code>	The index of the element for which data should be accessed.
--	---

Returns

Read/write reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Referenced by `std::match_results< _Bi_iter, _Alloc >::operator[]()`.

pop_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::pop_back ( ) [inline], [constexpr], [noexcept]
```

Removes last element.

This is a typical stack operation. It shrinks the vector by one.


Note that no data is returned, and if the last element's data is needed, it should be retrieved before `pop_back()` is called.

push_back()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::push_back (
    const value_type & __x ) [inline], [constexpr]
```

Add data to the end of the vector.

Parameters

 <code>__x</code>	Data to be added.
--	-------------------

This is a typical stack operation. The function creates an element at the end of the vector and assigns the given data to it. Due to the nature of a vector this operation can be done in constant time if the vector has preallocated space available.

References `std::vector< _Tp, _Alloc >::end()`.

rbegin() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
```

```
constexpr const_reverse_iterator std::vector<_Tp, _Alloc >::rbegin ( ) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

References [std::vector<_Tp, _Alloc >::end\(\)](#).

rbegin() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr reverse_iterator std::vector<_Tp, _Alloc >::rbegin ( ) [inline], [constexpr], [noexcept]
```

Returns a read/write reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

References [std::vector<_Tp, _Alloc >::end\(\)](#).

rend() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr const_reverse_iterator std::vector<_Tp, _Alloc >::rend ( ) const [inline], [constexpr], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

References [std::vector<_Tp, _Alloc >::begin\(\)](#).

rend() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr reverse_iterator std::vector<_Tp, _Alloc >::rend ( ) [inline], [constexpr], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

References [std::vector<_Tp, _Alloc >::begin\(\)](#).

reserve()

```
template<typename _Tp , typename _Alloc >
constexpr void vector::reserve (
    size_type __n ) [constexpr]
```

Attempt to preallocate enough memory for specified number of elements.

Parameters

_↵ _n	Number of elements required.
--	------------------------------

Exceptions

std::length_error	If <i>n</i> exceeds <code>max_size()</code> .
-----------------------------------	---

This function attempts to reserve enough memory for the vector to hold the specified number of elements. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the number of elements that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of vector data.

References [std::_Destroy\(\)](#), and [std::size\(\)](#).

resize() [1/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::resize (
    size_type __new_size ) [inline], [constexpr]
```

Resizes the vector to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the vector should contain.
-------------------------	---

This function will resize the vector to the specified number of elements. If the number is smaller than the vector's current size the vector is truncated, otherwise default constructed elements are appended.

References [std::vector< _Tp, _Alloc >::size\(\)](#).

Referenced by [__gnu_parallel::__shrink_and_double\(\)](#), and [__gnu_parallel::multiway_merge_exact_splitting\(\)](#).

resize() [2/2]

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::resize (
    size_type __new_size,
    const value_type & __x ) [inline], [constexpr]
```

Resizes the vector to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the vector should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the vector to the specified number of elements. If the number is smaller than the vector's current size the vector is truncated, otherwise the vector is extended and new elements are populated with given data.

References [std::vector< _Tp, _Alloc >::end\(\)](#), and [std::vector< _Tp, _Alloc >::size\(\)](#).

shrink_to_fit()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr void std::vector< _Tp, _Alloc >::shrink_to_fit ( ) [inline], [constexpr]
```

A non-binding request to reduce capacity() to size().

size()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
constexpr size_type std::vector< _Tp, _Alloc >::size ( ) const [inline], [constexpr], [noexcept]
```

Returns the number of elements in the vector.

Referenced by [__gnu_parallel::__shrink\(\)](#), [__gnu_parallel::__shrink_and_double\(\)](#), [std::vector< _Tp, _Alloc >::M_range_check\(\)](#), [std::match_results< _Bi_iter, _Alloc >::empty\(\)](#), [__gnu_parallel::list_partition\(\)](#), [std::discrete_distribution< _IntType >::max\(\)](#), [std::vector< _Tp, _Alloc >::operator=\(\)](#), [std::operator==\(\)](#), [std::vector< _Tp, _Alloc >::resize\(\)](#), [std::vector< _Tp, _Alloc >::resize\(\)](#), and [std::match_results< _Bi_iter, _Alloc >::size\(\)](#).

swap()

```
template<typename _Tp , typename _Alloc = std::allocator<_Tp>>
```

```
constexpr void std::vector< _Tp, _Alloc >::swap (
    vector< _Tp, _Alloc > & __x ) [inline], [constexpr], [noexcept]
```

Swaps data with another vector.

Parameters

<code>__x</code>	A vector of the same element and allocator types.
------------------	---

This exchanges the elements between two vectors in constant time. (Three pointers, so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(v1,v2)` will feed to this function.

Whether the allocators are swapped depends on the allocator traits.

Referenced by [std::match_results< _Bi_iter, _Alloc >::swap\(\)](#).

The documentation for this class was generated from the following files:

- [stl_vector.h](#)
- [vector.tcc](#)

6.1057 std::vector< bool, _Alloc > Class Template Reference

```
#include <stl_bvector.h>
```

Inherits `std::_Bvector_base< _Alloc >`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Bit_const_iterator` **const_iterator**
- typedef `const bool *` **const_pointer**
- typedef `bool` **const_reference**
- typedef [std::reverse_iterator](#)< `const_iterator` > **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `_Bit_iterator` **iterator**
- typedef `_Bit_reference *` **pointer**
- typedef `_Bit_reference` **reference**
- typedef [std::reverse_iterator](#)< `iterator` > **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `bool` **value_type**

Public Member Functions

- template<typename `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>>`
constexpr **vector** (`_InputIterator` __first, `_InputIterator` __last, `const allocator_type &__a=allocator_type()`)
- constexpr **vector** (`const allocator_type &__a`)
- constexpr **vector** (`const vector &__x`)
- constexpr **vector** (`const vector &__x`, `const __type_identity_t< allocator_type > &__a`)
- constexpr **vector** ([initializer_list](#)< `bool` > __l, `const allocator_type &__a=allocator_type()`)
- constexpr **vector** (`size_type __n`, `const allocator_type &__a=allocator_type()`)
- constexpr **vector** (`size_type __n`, `const bool &__value`, `const allocator_type &__a=allocator_type()`)
- **vector** ([vector](#) &&)=default
- constexpr **vector** ([vector](#) &&__x, `const __type_identity_t< allocator_type > &__a`) noexcept(`_Bit_alloc_traits::__S_always_equal()`)
- template<typename `_InputIterator` , typename = `std::_RequireInputIter<_InputIterator>>`
constexpr void **assign** (`_InputIterator` __first, `_InputIterator` __last)

- constexpr void **assign** (initializer_list< bool > __l)
- constexpr void **assign** (size_type __n, const bool &__x)
- constexpr reference **at** (size_type __n)
- constexpr const_reference **at** (size_type __n) const
- constexpr reference **back** ()
- constexpr const_reference **back** () const
- constexpr const_iterator **begin** () const noexcept
- constexpr iterator **begin** () noexcept
- constexpr size_type **capacity** () const noexcept
- constexpr const_iterator **cbegin** () const noexcept
- constexpr const_iterator **cend** () const noexcept
- constexpr void **clear** () noexcept
- constexpr const_reverse_iterator **crbegin** () const noexcept
- constexpr const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
constexpr iterator **emplace** (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
constexpr reference **emplace_back** (_Args &&... __args)
- constexpr bool **empty** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr iterator **end** () noexcept
- constexpr iterator **erase** (const_iterator __first, const_iterator __last)
- constexpr iterator **erase** (const_iterator __position)
- constexpr void **flip** () noexcept
- constexpr reference **front** ()
- constexpr const_reference **front** () const
- constexpr allocator_type **get_allocator** () const
- constexpr iterator **insert** (const_iterator __p, initializer_list< bool > __l)
- iterator **insert** (const_iterator __position)
- template<typename _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
constexpr iterator **insert** (const_iterator __position, _InputIterator __first, _InputIterator __last)
- constexpr iterator **insert** (const_iterator __position, const bool &__x)
- constexpr iterator **insert** (const_iterator __position, size_type __n, const bool &__x)
- constexpr size_type **max_size** () const noexcept
- constexpr vector & **operator=** (const vector &__x)
- constexpr vector & **operator=** (initializer_list< bool > __l)
- constexpr vector & **operator=** (vector &&__x) noexcept(_Bit_alloc_traits::_S_nothrow_move())
- constexpr reference **operator[]** (size_type __n)
- constexpr const_reference **operator[]** (size_type __n) const
- constexpr void **pop_back** ()
- constexpr void **push_back** (bool __x)
- constexpr const_reverse_iterator **rbegin** () const noexcept
- constexpr reverse_iterator **rbegin** () noexcept
- constexpr const_reverse_iterator **rend** () const noexcept
- constexpr reverse_iterator **rend** () noexcept
- constexpr void **reserve** (size_type __n)
- constexpr void **resize** (size_type __new_size, bool __x=bool())
- constexpr void **shrink_to_fit** ()
- constexpr size_type **size** () const noexcept
- constexpr void **swap** (vector &__x) noexcept

Static Public Member Functions

- static constexpr void **swap** (reference __x, reference __y) noexcept

Protected Types

- typedef [__gnu_cxx::__alloc_traits](#)< _Alloc >::template rebind< _Bit_type >::other **_Bit_alloc_type**

Protected Member Functions

- constexpr _Bit_pointer **_M_allocate** (size_t __n)
- template<typename _ForwardIterator >
constexpr void **_M_assign_aux** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _InputIterator >
constexpr void **_M_assign_aux** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- constexpr size_type **_M_check_len** (size_type __n, const char *__s) const
- constexpr iterator **_M_copy_aligned** (const_iterator __first, const_iterator __last, iterator __result)
- constexpr void **_M_deallocate** ()
- constexpr iterator **_M_erase** (iterator __first, iterator __last)
- constexpr iterator **_M_erase** (iterator __pos)
- constexpr void **_M_erase_at_end** (iterator __pos)
- constexpr void **_M_fill_assign** (size_t __n, bool __x)
- constexpr void **_M_fill_insert** (iterator __position, size_type __n, bool __x)
- constexpr const _Bit_alloc_type & **_M_get_Bit_allocator** () const noexcept
- constexpr _Bit_alloc_type & **_M_get_Bit_allocator** () noexcept
- constexpr void **_M_initialize** (size_type __n)
- template<typename _ForwardIterator >
constexpr void **_M_initialize_range** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _InputIterator >
constexpr void **_M_initialize_range** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- constexpr void **_M_initialize_value** (bool __x) noexcept
- constexpr void **_M_insert_aux** (iterator __position, bool __x)
- template<typename _InputIterator >
constexpr void **_M_insert_range** (iterator __pos, _InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
constexpr void **_M_insert_range** (iterator __position, _ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- constexpr void **_M_move_data** (_Bvector_base &&__x) noexcept
- constexpr void **_M_range_check** (size_type __n) const
- constexpr void **_M_reallocate** (size_type __n)
- constexpr bool **_M_shrink_to_fit** ()
- void **data** ()=delete

Static Protected Member Functions

- static constexpr size_t **_S_nword** (size_t __n)

Protected Attributes

- _Bvector_impl **_M_impl**

Friends

- struct [std::hash](#)< **vector** >

6.1057.1 Detailed Description

```
template<typename _Alloc>
class std::vector< bool, _Alloc >
```

A specialization of vector for booleans which offers fixed time access to individual elements in any order.

Template Parameters

<code>_Alloc</code>	Allocator type.
---------------------	-----------------

Note that `vector<bool>` does not actually meet the requirements for being a container. This is because the reference and pointer types are not really references and pointers to `bool`. See DR96 for details.

See also

[vector](#) for function documentation.

In some terminology a vector can be described as a dynamic C-style array, it offers fast and efficient access to individual elements in any order and saves the user from worrying about memory and size allocation. Subscripting (`[]`) access is also provided as with C-style arrays.

The documentation for this class was generated from the following files:

- [stl_bvector.h](#)
- [vector.tcc](#)

6.1058 std::ranges::view_base Struct Reference

```
#include <ranges_base.h>
```

6.1058.1 Detailed Description

[range.view] The `ranges::view_base` type.

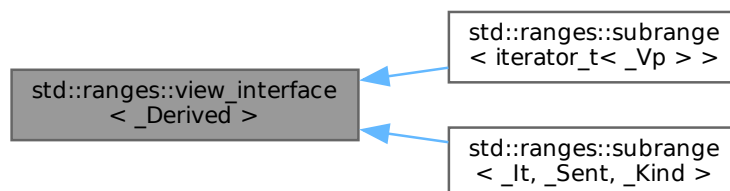
The documentation for this struct was generated from the following file:

- [ranges_base.h](#)

6.1059 std::ranges::view_interface< _Derived > Class Template Reference

```
#include <ranges_util.h>
```

Inheritance diagram for `std::ranges::view_interface< _Derived >`:



Public Member Functions

- constexpr decltype(auto) **back** ()
- constexpr decltype(auto) **back** () const
- constexpr auto **data** () const noexcept(noexcept(ranges::begin(_M_derived())))
- constexpr auto **data** () noexcept(noexcept(ranges::begin(_M_derived())))
- constexpr bool **empty** () const noexcept(noexcept(_S_empty(_M_derived())))
- constexpr bool **empty** () noexcept(noexcept(_S_empty(_M_derived())))
- constexpr decltype(auto) **front** ()
- constexpr decltype(auto) **front** () const
- constexpr **operator bool** () const noexcept(noexcept(ranges::empty(_M_derived())))
- constexpr **operator bool** () noexcept(noexcept(ranges::empty(_M_derived())))
- template<random_access_range _Range = _Derived>
constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n)
- template<random_access_range _Range = const _Derived>
constexpr decltype(auto) **operator[]** (range_difference_t< _Range > __n) const
- constexpr auto **size** () const noexcept(noexcept(_S_size(_M_derived())))
- constexpr auto **size** () noexcept(noexcept(_S_size(_M_derived())))

6.1059.1 Detailed Description

```
template<typename _Derived>
requires is_class_v<_Derived> && same_as<_Derived, remove_cv_t<_Derived>>
class std::ranges::view_interface< _Derived >
```

The ranges::view_interface class template.

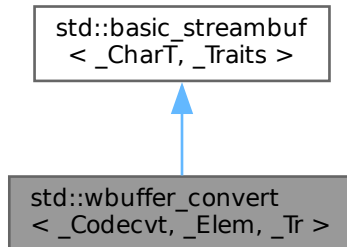
The documentation for this class was generated from the following files:

- [ranges_base.h](#)
- [ranges_util.h](#)

6.1060 std::wbuffer_convert< _Codecvt, _Elem, _Tr > Class Template Reference

```
#include <locale_conv.h>
```

Inheritance diagram for std::wbuffer_convert< _Codecvt, _Elem, _Tr >:



Public Types

- typedef `_Codecvt::state_type` **state_type**
- typedef `_CharT` **char_type**
- typedef `_Traits` **traits_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `traits_type::off_type` **off_type**
- typedef `basic_streambuf< char_type, traits_type > __streambuf_type`

Public Member Functions

- **wbuffer_convert** ()
- **wbuffer_convert** (const **wbuffer_convert** &)=delete
- **wbuffer_convert** (**streambuf** * __bytebuf, `_Codecvt` * __pcvt=new `_Codecvt`, `state_type` __state=state_type())
- **locale getloc** () const
- **streamsize in_avail** ()
- **wbuffer_convert & operator=** (const **wbuffer_convert** &)=delete
- **locale pubimbue** (const **locale** & __loc)
- **streambuf * rdbuf** () const noexcept
- **streambuf * rdbuf** (**streambuf** * __bytebuf) noexcept
- **int_type sbumpc** ()
- **int_type sgetc** ()
- **streamsize sgetn** (**char_type** * __s, **streamsize** __n)
- **int_type snextc** ()
- **int_type sputbackc** (**char_type** __c)
- **int_type sputc** (**char_type** __c)
- **streamsize sputn** (const **char_type** * __s, **streamsize** __n)
- **state_type state** () const noexcept
- **int_type sungetc** ()
- **basic_streambuf * pubsetbuf** (**char_type** * __s, **streamsize** __n)
- **pos_type pubseekoff** (**off_type** __off, `ios_base::seekdir` __way, `ios_base::openmode` __mode=`ios_base::in|ios_base::out`)
- **pos_type pubseekpos** (**pos_type** __sp, `ios_base::openmode` __mode=`ios_base::in|ios_base::out`)
- **int pubsync** ()

Protected Member Functions

- void **__safe_gbump** (**streamsize** __n)
- void **__safe_pbump** (**streamsize** __n)
- void **gbump** (int __n)
- virtual void **imbue** (const **locale** & __loc)
- **_Wide_streambuf::int_type overflow** (typename **_Wide_streambuf::int_type** __out)
- virtual **int_type pbackfail** (**int_type** __c=`traits_type::eof()`)
- void **pbump** (int __n)
- virtual **pos_type seekoff** (**off_type**, `ios_base::seekdir`, `ios_base::openmode`=`ios_base::in|ios_base::out`)
- virtual **pos_type seekpos** (**pos_type**, `ios_base::openmode`=`ios_base::in|ios_base::out`)
- virtual **basic_streambuf< char_type, _Traits > * setbuf** (**char_type** *, **streamsize**)

- void `setg` (`char_type` * __gbeg, `char_type` * __gnext, `char_type` * __gend)
- void `setp` (`char_type` * __pbeg, `char_type` * __pend)
- virtual `streamsize showmanyc` ()
- void `swap` (`basic_streambuf` & __sb)
- int `sync` ()
- virtual `int_type uflow` ()
- `_Wide_streambuf::int_type underflow` ()
- virtual `streamsize xsgetn` (`char_type` * __s, `streamsize` __n)
- virtual `streamsize xspn` (const `char_type` * __s, `streamsize` __n)
- `streamsize xspn` (const typename `_Wide_streambuf::char_type` * __s, `streamsize` __n)
- `char_type` * `eback` () const
- `char_type` * `gptr` () const
- `char_type` * `egptr` () const
- `char_type` * `pbase` () const
- `char_type` * `pptr` () const
- `char_type` * `eptr` () const

Protected Attributes

- `locale _M_buf_locale`
- `char_type` * `_M_in_beg`
- `char_type` * `_M_in_cur`
- `char_type` * `_M_in_end`
- `char_type` * `_M_out_beg`
- `char_type` * `_M_out_cur`
- `char_type` * `_M_out_end`

6.1060.1 Detailed Description

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
class std::wbuffer_convert<_Codecvt, _Elem, _Tr>
```

Buffer conversions.

6.1060.2 Member Typedef Documentation

`__streambuf_type`

```
template<typename _CharT, typename _Traits>
typedef basic_streambuf<char_type, traits_type> std::basic_streambuf<_CharT, _Traits>::__streambuf_type [inherited]
```

This is a non-standard type.

`char_type`

```
template<typename _CharT, typename _Traits>
typedef _CharT std::basic_streambuf<_CharT, _Traits>::char_type [inherited]
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

int_type

```
template<typename _CharT , typename _Traits >
```

```
typedef traits_type::int_type std::basic_streambuf< _CharT, _Traits >::int_type [inherited]
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

off_type

```
template<typename _CharT , typename _Traits >
```

```
typedef traits_type::off_type std::basic_streambuf< _CharT, _Traits >::off_type [inherited]
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

pos_type

```
template<typename _CharT , typename _Traits >
```

```
typedef traits_type::pos_type std::basic_streambuf< _CharT, _Traits >::pos_type [inherited]
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

traits_type

```
template<typename _CharT , typename _Traits >
```

```
typedef _Traits std::basic_streambuf< _CharT, _Traits >::traits_type [inherited]
```

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

6.1060.3 Constructor & Destructor Documentation**wbuffer_convert() [1/2]**

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
```

```
std::wbuffer_convert< _Codecvt, _Elem, _Tr >::wbuffer_convert ( ) [inline]
```

Default constructor.

wbuffer_convert() [2/2]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
```

```
std::wbuffer_convert< _Codecvt, _Elem, _Tr >::wbuffer_convert (
```

```
    streambuf * __bytebuf,
```

```
    _Codecvt * __pcvt = new _Codecvt,
```

```
    state_type __state = state_type() ) [inline], [explicit]
```

Constructor.

Parameters

<code>__bytebuf</code>	The underlying byte stream buffer.
<code>__pcvt</code>	The facet to use for conversions.
<code>__state</code>	Initial conversion state.

Takes ownership of `__pcvt` and will delete it in the destructor.

References `std::basic_streambuf< _CharT, _Traits >::setg()`, and `std::basic_streambuf< _CharT, _Traits >::setp()`.

6.1060.4 Member Function Documentation

`eback()`

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::eback ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

`egptr()`

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::egptr ( ) const [inline], [protected],
[inherited]
```

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow()`.

`epptr()`

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::epptr ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

`gbump()`

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::gbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

getloc()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::getloc ( ) const [inline], [inherited]
Locale access.
```

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

gptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::gptr ( ) const [inline], [protected], [inherited]
Access to the get area.
```

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow\(\)](#).

imbue()

```
template<typename _CharT , typename _Traits >
virtual void std::basic_streambuf< _CharT, _Traits >::imbue (
    const locale & __loc ) [inline], [protected], [virtual], [inherited]
```

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >](#), [std::basic_filebuf< _CharT, std::char_traits< _CharT > >](#), and [std::basic_filebuf< char_type, traits_type >](#).

in_avail()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::in_avail ( ) [inline], [inherited]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

overflow()

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
_Wide_streambuf::int_type std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow (
    typename _Wide_streambuf::int_type __c) [inline], [protected], [virtual]
```

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::basic_streambuf<_CharT, _Traits>::sputc()`.

pbackfail()

```
template<typename _CharT, typename _Traits>
virtual int_type std::basic_streambuf<_CharT, _Traits>::pbackfail (
    int_type __c = traits_type::eof()) [inline], [protected], [virtual], [inherited]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns eof().

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits, _Alloc>` and `std::basic_stringbuf<_CharT, _Traits, _Alloc>`.

pbase()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pbase ( ) const [inline], [protected],
[inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pbump()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::pbump (
    int __n ) [inline], [protected], [inherited]
```

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

pptr()

```
template<typename _CharT , typename _Traits >
char_type * std::basic_streambuf< _CharT, _Traits >::pptr ( ) const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

pubimbue()

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::pubimbue (
    const locale & __loc ) [inline], [inherited]
```

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

pubseekoff()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off,
    ios_base::seekdir __way,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

pubseekpos()

```
template<typename _CharT , typename _Traits >
pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (
    pos_type __sp,
    ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekpos function.

pubsetbuf()

```
template<typename _CharT , typename _Traits >
basic_streambuf * std::basic_streambuf< _CharT, _Traits >::pubsetbuf (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of pubfoo dispatch to the protected derived foo member functions, passing the arguments (if any) and returning the result unchanged.

pubsync()

```
template<typename _CharT , typename _Traits >
int std::basic_streambuf< _CharT, _Traits >::pubsync ( ) [inline], [inherited]
```

Calls virtual sync function.

Referenced by [std::wbuffer_convert< _Codecvt, _Elem, _Tr >::sync\(\)](#), and [std::basic_istream< _CharT, _Traits >::sync\(\)](#).

sbumpc()

```
template<typename _CharT , typename _Traits >
int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns [uflow\(\)](#).

Referenced by [std::basic_istream< _CharT, _Traits >::ignore\(\)](#), [std::basic_istream< char >::ignore\(\)](#), [std::istreambuf_iterator< _CharT, _Traits >::operator++\(\)](#), and [std::basic_istream< char >::seekg\(\)](#).

seekoff()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekoff (
    off_type ,
    ios_base::seekdir ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >](#), [std::basic_filebuf< _CharT, std::char_traits< _CharT > >](#), [std::basic_filebuf< char_type, traits_type >](#), and [std::basic_stringbuf< _CharT, traits_type, _Alloc >](#).

seekpos()

```
template<typename _CharT , typename _Traits >
virtual pos_type std::basic_streambuf< _CharT, _Traits >::seekpos (
    pos_type ,
    ios_base::openmode = ios_base::in | ios_base::out ) [inline], [protected], [virtual],
[inherited]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >](#), [std::basic_filebuf< _CharT, std::char_traits< _CharT > >](#), [std::basic_filebuf< char_type, traits_type >](#), and [std::basic_stringbuf< _CharT, traits_type, _Alloc >](#).

setbuf()

```
template<typename _CharT , typename _Traits >
virtual basic_streambuf< char_type, _Traits > * std::basic_streambuf< _CharT, _Traits >::setbuf
(
    char_type * ,
    streamsize ) [inline], [protected], [virtual], [inherited]
```

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >`, `std::basic_filebuf< _CharT, std::char_traits< _CharT > >`, `std::basic_filebuf< char_type, traits_type >`, and `std::basic_stringbuf< _CharT, traits_type, allocator_type >`.

setg()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setg (
    char_type * __gbeg,
    char_type * __gnext,
    char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Referenced by `std::wbuffer_convert<_Codecvt, _Elem, _Tr >::wbuffer_convert()`.

setp()

```
template<typename _CharT , typename _Traits >
void std::basic_streambuf< _CharT, _Traits >::setp (
    char_type * __pbeg,
    char_type * __pend ) [inline], [protected], [inherited]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == ep_ptr()`

Referenced by `std::wbuffer_convert<_Codecv, _Elem, _Tr>::wbuffer_convert()`.

sgetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sgetc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Referenced by `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::istreambuf_iterator<_CharT, _Traits>::operator++()`, and `std::basic_istream<char>::seekg()`.

sgetn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (
    char_type * __s,
    streamsize __n ) [inline], [inherited]
```

Entry point for `xsggetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsggetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

showmanyc()

```
template<typename _CharT, typename _Traits>
virtual streamsize std::basic_streambuf<_CharT, _Traits>::showmanyc ( ) [inline], [protected],
[virtual], [inherited]
```

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to underflow or uflow] will not return eof() but that they will return immediately.*

The standard adds that *the morphemes of showmanyc are **es-how-many-see**, not **show-manic**.*

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<_CharT, std::char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits>](#).

snextc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::snextc ( ) [inline], [inherited]
```

Getting the next character.

Returns

The next character, or eof.

Calls `sputc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Referenced by [std::basic_istream<_CharT, _Traits>::sentry::sentry\(\)](#), [std::basic_istream<_CharT, _Traits>::get\(\)](#), [std::basic_istream<char>::seekg\(\)](#), and [std::basic_istream<char>::unget\(\)](#).

sputbackc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
    char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Referenced by [std::basic_istream<_CharT, _Traits>::putback\(\)](#).

sputc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sputc (
    char_type __c ) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

__c, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores __c in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(↵__c)`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow()`.

sputn()

```
template<typename _CharT, typename _Traits>
streamsize std::basic_streambuf<_CharT, _Traits>::sputn (
    const char_type * __s,
    streamsize __n) [inline], [inherited]
```

Entry point for all single-character output functions.

Parameters

↵ __s	A buffer read area.
↵ __n	A count.

One of two public output functions.

Returns `xsputn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

state()

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
state_type std::wbuffer_convert<_Codecvt, _Elem, _Tr>::state ( ) const [inline], [noexcept]
```

The conversion state following the last conversion.

sungetc()

```
template<typename _CharT, typename _Traits>
int_type std::basic_streambuf<_CharT, _Traits>::sungetc ( ) [inline], [inherited]
```

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

sync()

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
int std::wbuffer_convert<_Codecvt, _Elem, _Tr>::sync ( ) [inline], [protected], [virtual]
```

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::basic_streambuf<_CharT, _Traits>::pubsync()`.

uflow()

```
template<typename _CharT , typename _Traits >
virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ( ) [inline], [protected], [virtual],
[inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

underflow()

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
_Wide_streambuf::int_type std::wbuffer_convert< _Codecvt, _Elem, _Tr >::underflow ( ) [inline],
[protected], [virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

References `std::basic_streambuf<_CharT, _Traits>::egptr()`, and `std::basic_streambuf<_CharT, _Traits>::gptr()`.

xsgetn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::xsgetn (
    char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

References `std::min()`.

xsputn()

```
template<typename _CharT , typename _Traits >
streamsize std::basic_streambuf< _CharT, _Traits >::xsputn (
    const char_type * __s,
    streamsize __n ) [protected], [virtual], [inherited]
```

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<_CharT, std::char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

References `std::min()`.

6.1060.5 Member Data Documentation**`_M_buf_locale`**

```
template<typename _CharT , typename _Traits >
locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale [protected], [inherited]
```

Current locale setting.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

`_M_in_beg`

```
template<typename _CharT , typename _Traits >
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_beg [protected], [inherited]
```

Start of get area.

_M_in_cur

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_cur [protected], [inherited]
```

Current read area.

_M_in_end

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_end [protected], [inherited]
```

End of get area.

_M_out_beg

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_beg [protected], [inherited]
```

Start of put area.

_M_out_cur

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_cur [protected], [inherited]
```

Current put area.

_M_out_end

```
template<typename _CharT, typename _Traits>
```

```
char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_end [protected], [inherited]
```

End of put area.

The documentation for this class was generated from the following file:

- [locale_conv.h](#)

6.1061 std::weak_ptr<_Tp> Class Template Reference

```
#include <memory>
```

Inherits std::__weak_ptr<_Tp, _Lp>.

Public Types

- using **element_type** = typename [remove_extent](#)<_Tp>::type

Public Member Functions

- template<typename _Yp, typename = _Constructible<const shared_ptr<_Yp>&>>>
weak_ptr (const [shared_ptr](#)<_Yp> &__r) noexcept
- **weak_ptr** (const [weak_ptr](#) &) noexcept=default
- template<typename _Yp, typename = _Constructible<const weak_ptr<_Yp>&>>>
weak_ptr (const [weak_ptr](#)<_Yp> &__r) noexcept
- **weak_ptr** ([weak_ptr](#) &&) noexcept=default
- template<typename _Yp, typename = _Constructible<weak_ptr<_Yp>>>>
weak_ptr ([weak_ptr](#)<_Yp> &&__r) noexcept

- bool **expired** () const noexcept
- [shared_ptr](#)< _Tp > **lock** () const noexcept
- template<typename _Yp >
_Assignable< const [shared_ptr](#)< _Yp > & > **operator=** (const [shared_ptr](#)< _Yp > &__r) noexcept
- [weak_ptr](#) & **operator=** (const [weak_ptr](#) &__r) noexcept=default
- template<typename _Yp >
_Assignable< const [weak_ptr](#)< _Yp > & > **operator=** (const [weak_ptr](#)< _Yp > &__r) noexcept
- [weak_ptr](#) & **operator=** ([weak_ptr](#) &&__r) noexcept=default
- template<typename _Yp >
_Assignable< [weak_ptr](#)< _Yp > > **operator=** ([weak_ptr](#)< _Yp > &&__r) noexcept
- template<typename _Tp1 >
bool **owner_before** (const __shared_ptr< _Tp1, _Lp > &__rhs) const noexcept
- template<typename _Tp1 >
bool **owner_before** (const __weak_ptr< _Tp1, _Lp > &__rhs) const noexcept
- void **reset** () noexcept
- void **swap** (__weak_ptr &__s) noexcept
- long **use_count** () const noexcept

Related Symbols

(Note that these are not member symbols.)

- template<typename _Tp >
void **swap** ([weak_ptr](#)< _Tp > &__a, [weak_ptr](#)< _Tp > &__b) noexcept

6.1061.1 Detailed Description

template<typename _Tp>
class std::weak_ptr< _Tp >

A non-owning observer for a pointer owned by a [shared_ptr](#).

Since

C++11

A [weak_ptr](#) provides a safe alternative to a raw pointer when you want a non-owning reference to an object that is managed by a [shared_ptr](#).

Unlike a raw pointer, a [weak_ptr](#) can be converted to a new [shared_ptr](#) that shares ownership with every other [shared_ptr](#) that already owns the pointer. In other words you can upgrade from a non-owning “weak” reference to an owning [shared_ptr](#), without having access to any of the existing [shared_ptr](#) objects.

Also unlike a raw pointer, a [weak_ptr](#) does not become “dangling” after the object it points to has been destroyed. Instead, a [weak_ptr](#) becomes *expired* and can no longer be converted to a [shared_ptr](#) that owns the freed pointer, so you cannot accidentally access the pointed-to object after it has been destroyed.

The documentation for this class was generated from the following file:

- [bits/shared_ptr.h](#)

6.1062 std::weibull_distribution< _RealType > Class Template Reference

```
#include <random.h>
```

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **weibull_distribution** (_RealType __a, _RealType __b=_RealType(1))
- **weibull_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **a** () const
- _RealType **b** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- bool **operator==** (const [weibull_distribution](#) &__d1, const [weibull_distribution](#) &__d2)

6.1062.1 Detailed Description

template<typename _RealType = double>
class std::weibull_distribution<_RealType>

A weibull_distribution random number distribution.

The formula for the normal probability density function is:

$$p(x|\alpha, \beta) = \frac{\alpha}{\beta} \left(\frac{x}{\beta}\right)^{\alpha-1} \exp\left(-\left(\frac{x}{\beta}\right)^{\alpha}\right)$$

6.1062.2 Member Typedef Documentation

result_type

```
template<typename _RealType = double>
typedef _RealType std::weibull\_distribution<_RealType>::result_type
```

The type of the range of the distribution.

6.1062.3 Member Function Documentation

a()

```
template<typename _RealType = double>
_RealType std::weibull\_distribution<_RealType>::a ( ) const [inline]
```

Return the *a* parameter of the distribution.

b()

```
template<typename _RealType = double>
_RealType std::weibull_distribution< _RealType >::b ( ) const [inline]
Return the b parameter of the distribution.
```

max()

```
template<typename _RealType = double>
result_type std::weibull_distribution< _RealType >::max ( ) const [inline]
Returns the least upper bound value of the distribution.
References std::numeric_limits< _Tp >::max().
```

min()

```
template<typename _RealType = double>
result_type std::weibull_distribution< _RealType >::min ( ) const [inline]
Returns the greatest lower bound value of the distribution.
```

operator()()

```
template<typename _RealType = double>
template<typename _UniformRandomNumberGenerator >
result_type std::weibull_distribution< _RealType >::operator() (
    _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

References `std::weibull_distribution< _RealType >::operator()()`.

Referenced by `std::weibull_distribution< _RealType >::operator()()`.

param() [1/2]

```
template<typename _RealType = double>
param_type std::weibull_distribution< _RealType >::param ( ) const [inline]
Returns the parameter set of the distribution.
Referenced by std::operator>>().
```

param() [2/2]

```
template<typename _RealType = double>
void std::weibull_distribution< _RealType >::param (
    const param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

reset()

```
template<typename _RealType = double>
void std::weibull_distribution< _RealType >::reset ( ) [inline]
Resets the distribution state.
```

6.1062.4 Friends And Related Symbol Documentation

operator==

```
template<typename _RealType = double>
bool operator== (
    const weibull_distribution< _RealType > & __d1,
    const weibull_distribution< _RealType > & __d2 ) [friend]
```

Return true if two Weibull distributions have the same parameters.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

6.1063 std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc > Class Template Reference

```
#include <locale_conv.h>
```

Public Types

- typedef [basic_string](#)< char, [char_traits](#)< char >, _Byte_alloc > **byte_string**
- typedef [wide_string::traits_type::int_type](#) **int_type**
- typedef [_Codecvt::state_type](#) **state_type**
- typedef [basic_string](#)< _Elem, [char_traits](#)< _Elem >, _Wide_alloc > **wide_string**

Public Member Functions

- [wstring_convert](#) ()
- [wstring_convert](#) (_Codecvt * __pcvt)
- [wstring_convert](#) (_Codecvt * __pcvt, state_type __state)
- [wstring_convert](#) (const [byte_string](#) & __byte_err, const [wide_string](#) & __wide_err=[wide_string](#)())
- [wstring_convert](#) (const [wstring_convert](#) &)=delete
- size_t [converted](#) () const noexcept
- [wstring_convert](#) & **operator=** (const [wstring_convert](#) &)=delete
- state_type [state](#) () const
- [wide_string from_bytes](#) (char __byte)
- [wide_string from_bytes](#) (const char * __ptr)
- [wide_string from_bytes](#) (const [byte_string](#) & __str)
- [wide_string from_bytes](#) (const char * __first, const char * __last)
- [byte_string to_bytes](#) (_Elem __wchar)
- [byte_string to_bytes](#) (const _Elem * __ptr)
- [byte_string to_bytes](#) (const [wide_string](#) & __wstr)
- [byte_string to_bytes](#) (const _Elem * __first, const _Elem * __last)

6.1063.1 Detailed Description

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, type-
name _Byte_alloc = allocator<char>>
```

```
class std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc >
```

String conversions.

6.1063.2 Constructor & Destructor Documentation

wstring_convert() [1/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::wstring_convert ( ) [inline]
Default constructor.
```

wstring_convert() [2/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::wstring_convert (
    _Codecvt * __pcvt ) [inline], [explicit]
```

Constructor.

Parameters

<code>__pcvt</code>	The facet to use for conversions.
---------------------	-----------------------------------

Takes ownership of `__pcvt` and will delete it in the destructor.

wstring_convert() [3/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::wstring_convert (
    _Codecvt * __pcvt,
    state_type __state ) [inline]
```

Construct with an initial conversion state.

Parameters

<code>__pcvt</code>	The facet to use for conversions.
<code>__state</code>	Initial conversion state.

Takes ownership of `__pcvt` and will delete it in the destructor. The object's conversion state will persist between conversions.

wstring_convert() [4/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::wstring_convert (
    const byte_string & __byte_err,
    const wide_string & __wide_err = wide_string() ) [inline], [explicit]
```

Construct with error strings.

Parameters

<code>__byte_err</code>	A string to return on failed conversions.
<code>__wide_err</code>	A wide string to return on failed conversions.

6.1063.3 Member Function Documentation

converted()

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
size_t std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::converted ( ) const
[inline], [noexcept]
```

The number of elements successfully converted in the last conversion.

from_bytes() [1/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
wstring std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes (
    char __byte ) [inline]
```

Convert from bytes.

References [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#).

Referenced by [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#), [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#), and [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#).

from_bytes() [2/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
wstring std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes (
    const byte_string & __str ) [inline]
```

Convert from bytes.

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#), and [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#).

from_bytes() [3/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
wstring std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes (
    const char * __first,
    const char * __last ) [inline]
```

Convert from bytes.

References [std::basic_string< _CharT, _Traits, _Alloc >::get_allocator\(\)](#).

from_bytes() [4/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
wstring std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes (
    const char * __ptr ) [inline]
```

Convert from bytes.

References [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes\(\)](#).

state()

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>
state_type std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::state ( ) const
[inline]
```


The final conversion state of the last conversion.

to_bytes() [1/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>>
byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes (
    _Elem __wchar ) [inline]
```

Convert to bytes.

References [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#).

Referenced by [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#), [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#), and [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#).

to_bytes() [2/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>>
byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes (
    const _Elem * __first,
    const _Elem * __last ) [inline]
```

Convert to bytes.

References [std::basic_string< _CharT, _Traits, _Alloc >::get_allocator\(\)](#).

to_bytes() [3/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>>
byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes (
    const _Elem * __ptr ) [inline]
```

Convert to bytes.

References [std::basic_string< _CharT, _Traits, _Alloc >::length\(\)](#), and [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#).

to_bytes() [4/4]

```
template<typename _Codecvt , typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>,
typename _Byte_alloc = allocator<char>>>
byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes (
    const wide_string & __wstr ) [inline]
```

Convert to bytes.

References [std::basic_string< _CharT, _Traits, _Alloc >::data\(\)](#), [std::basic_string< _CharT, _Traits, _Alloc >::size\(\)](#),

and [std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes\(\)](#).

The documentation for this class was generated from the following file:

- [locale_conv.h](#)

7 File Documentation

7.1 compare File Reference

Classes

- struct [std::compare_three_way_result< _Tp, _Up >](#)

Namespaces

- namespace `std`
- namespace `std::__detail`

Macros

- `#define __cpp_lib_three_way_comparison`
- `#define _COMPARE`

Typedefs

- `template<typename _Tp, typename _Up >`
`using std::__detail::__cmp3way_res_t = decltype(std::declval<_Tp >() <= > std::declval<_Up >())`
- `template<typename _Tp, typename _Up = _Tp>`
`using std::__detail::__synth3way_t = decltype(__detail::__synth3way(std::declval<_Tp & >(), std::declval<_Up & >()))`
- `template<typename... _Ts>`
`using std::common_comparison_category_t = typename common_comparison_category<_Ts... >::type`
- `template<typename _Tp, typename _Up = _Tp>`
`using std::compare_three_way_result_t = typename __detail::__cmp3way_res_impl<_Tp, _Up >::type`
- `using std::__cmp_cat::type = signed char`

Enumerations

- `enum class _Ncmp : type { _Unordered }`
- `enum class _Ord : type { equivalent , less , greater }`

Functions

- `template<typename... _Ts>`
`constexpr auto std::__detail::__common_cmp_cat ()`
- `template<floating_point _Tp>`
`constexpr weak_ordering std::__cmp_cust::__fp_weak_ordering (_Tp __e, _Tp __f)`
- `constexpr bool std::is_eq (partial_ordering __cmp) noexcept`
- `constexpr bool std::is_gt (partial_ordering __cmp) noexcept`
- `constexpr bool std::is_gteq (partial_ordering __cmp) noexcept`
- `constexpr bool std::is_lt (partial_ordering __cmp) noexcept`
- `constexpr bool std::is_lteq (partial_ordering __cmp) noexcept`
- `constexpr bool std::is_neq (partial_ordering __cmp) noexcept`

Variables

- `template<typename _Tp >`
`constexpr unsigned std::__detail::__cmp_cat_id`
- `template<> constexpr unsigned std::__detail::__cmp_cat_id< partial_ordering >`
- `template<> constexpr unsigned std::__detail::__cmp_cat_id< strong_ordering >`
- `template<> constexpr unsigned std::__detail::__cmp_cat_id< weak_ordering >`
- `constexpr struct std::__detail::__Synth3way std::__detail::__synth3way`
- `constexpr __cmp_cust::Partial_fallback std::compare_partial_order_fallback`
- `constexpr __cmp_cust::Strong_fallback std::compare_strong_order_fallback`
- `constexpr __cmp_cust::Weak_fallback std::compare_weak_order_fallback`
- `constexpr __cmp_cust::Partial_order std::partial_order`
- `constexpr __cmp_cust::Strong_order std::strong_order`
- `constexpr __cmp_cust::Weak_order std::weak_order`

7.1.1 Detailed Description

This is a Standard C++ Library header.

7.2 cxxabi.h File Reference

Classes

- class [__gnu_cxx::recursive_init_error](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [abi](#)

Typedefs

- typedef [__cxa_cdtor_return_type](#)(* [__cxxabiv1::__cxa_cdtor_type](#)) (void *)

Functions

- [__cxa_dependent_exception](#) * [__cxxabiv1::__cxa_allocate_dependent_exception](#) () noexcept
- int [__cxxabiv1::__cxa_atexit](#) (void*)(void *), void *, void *) noexcept
- void [__cxxabiv1::__cxa_bad_cast](#) ()
- void [__cxxabiv1::__cxa_bad_typeid](#) ()
- void * [__cxxabiv1::__cxa_begin_catch](#) (void *) noexcept
- [std::type_info](#) * [__cxxabiv1::__cxa_current_exception_type](#) () noexcept
- void [__cxxabiv1::__cxa_deleted_virtual](#) (void)
- char * [__cxxabiv1::__cxa_demangle](#) (const char * __mangled_name, char * __output_buffer, size_t * __length, int * __status)
- void [__cxxabiv1::__cxa_end_catch](#) ()
- void [__cxxabiv1::__cxa_finalize](#) (void *)
- void [__cxxabiv1::__cxa_free_dependent_exception](#) ([__cxa_dependent_exception](#) *) noexcept
- void [__cxxabiv1::__cxa_free_exception](#) (void *) noexcept
- void * [__cxxabiv1::__cxa_get_exception_ptr](#) (void *) noexcept
- [__cxa_eh_globals](#) * [__cxxabiv1::__cxa_get_globals](#) () noexcept
- [__cxa_eh_globals](#) * [__cxxabiv1::__cxa_get_globals_fast](#) () noexcept
- void [__cxxabiv1::__cxa_guard_abort](#) ([__guard](#) *) noexcept
- int [__cxxabiv1::__cxa_guard_acquire](#) ([__guard](#) *)
- void [__cxxabiv1::__cxa_guard_release](#) ([__guard](#) *) noexcept
- void [__cxxabiv1::__cxa_pure_virtual](#) (void)
- void [__cxxabiv1::__cxa_rethrow](#) ()
- int [__cxxabiv1::__cxa_thread_atexit](#) (void*)(void *), void *, void *) noexcept
- void [__cxxabiv1::__cxa_throw](#) (void *, [std::type_info](#) *, void*)(void *)
- void [__cxxabiv1::__cxa_throw_bad_array_new_length](#) ()
- [__cxa_vec_ctor_return_type](#) [__cxxabiv1::__cxa_vec_ctor](#) (void * __dest_array, void * __src_array, size_t __element_count, size_t __element_size, [__cxa_cdtor_return_type](#)(* __constructor)(void *, void *), [__cxa_cdtor_type](#) __destructor)
- void [__cxxabiv1::__cxa_vec_cleanup](#) (void * __array_address, size_t __element_count, size_t __s, [__cxa_cdtor_type](#) __destructor) noexcept
- [__cxa_vec_ctor_return_type](#) [__cxxabiv1::__cxa_vec_ctor](#) (void * __array_address, size_t __element_count, size_t __element_size, [__cxa_cdtor_type](#) __constructor, [__cxa_cdtor_type](#) __destructor)
- void [__cxxabiv1::__cxa_vec_delete](#) (void * __array_address, size_t __element_size, size_t __padding_size, [__cxa_cdtor_type](#) __destructor)

- void `__cxxabiv1::__cxa_vec_delete2` (void * __array_address, size_t __element_size, size_t __padding_size, __cxa_cdtor_type __destructor, void(* __dealloc)(void *))
- void `__cxxabiv1::__cxa_vec_delete3` (void * __array_address, size_t __element_size, size_t __padding_size, __cxa_cdtor_type __destructor, void(* __dealloc)(void *, size_t))
- void `__cxxabiv1::__cxa_vec_dtor` (void * __array_address, size_t __element_count, size_t __element_size, \leftrightarrow __cxa_cdtor_type __destructor)
- void * `__cxxabiv1::__cxa_vec_new` (size_t __element_count, size_t __element_size, size_t __padding_size, __cxa_cdtor_type __constructor, __cxa_cdtor_type __destructor)
- void * `__cxxabiv1::__cxa_vec_new2` (size_t __element_count, size_t __element_size, size_t __padding_size, __cxa_cdtor_type __constructor, __cxa_cdtor_type __destructor, void (* __alloc)(size_t), void(* __dealloc)(void *))
- void * `__cxxabiv1::__cxa_vec_new3` (size_t __element_count, size_t __element_size, size_t __padding_size, __cxa_cdtor_type __constructor, __cxa_cdtor_type __destructor, void (* __alloc)(size_t), void(* __dealloc)(void *, size_t))
- void * `__cxxabiv1::__dynamic_cast` (const void * __src_ptr, const __class_type_info * __src_type, const \leftrightarrow class_type_info * __dst_type, ptrdiff_t __src2dst)

7.2.1 Detailed Description

The header provides an interface to the C++ ABI.

7.2.2 Function Documentation

`__cxa_demangle()`

```
char * __cxxabiv1::__cxa_demangle (
    const char * __mangled_name,
    char * __output_buffer,
    size_t * __length,
    int * __status )
```

Demangling routine. ABI-mandated entry point in the C++ runtime library for demangling.

Parameters

<code>__mangled_name</code>	A NUL-terminated character string containing the name to be demangled.
<code>__output_buffer</code>	A region of memory, allocated with malloc, of * <code>__length</code> bytes, into which the demangled name is stored. If <code>__output_buffer</code> is not long enough, it is expanded using realloc. <code>__output_buffer</code> may instead be NULL; in that case, the demangled name is placed in a region of memory allocated with malloc.
<code>__length</code>	If <code>__length</code> is non-null, the length of the buffer containing the demangled name is placed in * <code>__length</code> .
<code>__status</code>	If <code>__status</code> is non-null, * <code>__status</code> is set to one of the following values: 0: The demangling operation succeeded. -1: A memory allocation failure occurred. -2: <code>mangled_name</code> is not a valid name under the C++ ABI mangling rules. -3: One of the arguments is invalid.

Returns

A pointer to the start of the NUL-terminated demangled name, or NULL if the demangling fails. The caller is responsible for deallocating this memory using `free`.

The demangling is performed using the C++ ABI mangling rules, with GNU extensions. For example, this function is used in `__gnu_cxx::__verbose_terminate_handler`.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/ext_demangling.html for other examples of use.

Note

The same demangling functionality is available via `libiberty` (`<libiberty/demangle.h>` and `libiberty.h`) in GCC 3.1 and later, but that requires explicit installation (`-enable-install-libiberty`) and uses a different API, although the ABI is unchanged.

7.3 exception File Reference

Classes

- class `std::bad_exception`

Namespaces

- namespace `__gnu_cxx`
- namespace `std`

Macros

- `#define __cpp_lib_uncaught_exceptions`
- `#define __EXCEPTION__`

Typedefs

- `typedef void(* std::terminate_handler) ()`
- `typedef void(* std::unexpected_handler) ()`

Functions

- `void __gnu_cxx::__verbose_terminate_handler ()`
- `terminate_handler std::get_terminate () noexcept`
- `unexpected_handler std::get_unexpected () noexcept`
- `terminate_handler std::set_terminate (terminate_handler) noexcept`
- `unexpected_handler std::set_unexpected (unexpected_handler) noexcept`
- `void std::terminate () noexcept`
- `bool std::uncaught_exception () noexcept`
- `int std::uncaught_exceptions () noexcept`
- `void std::unexpected ()`

7.3.1 Detailed Description

This is a Standard C++ Library header.

7.4 initializer_list File Reference

Classes

- class `std::initializer_list<_E>`

Namespaces

- namespace `std`

7.4.1 Detailed Description

This is a Standard C++ Library header.

7.5 new File Reference

Classes

- class `std::bad_alloc`
- struct `std::destroying_delete_t`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_laundry`
- `#define _NEW`

Typedefs

- typedef void(* `std::new_handler`) ()

Enumerations

- enum class `align_val_t` : `size_t`

Functions

- `new_handler std::get_new_handler` () noexcept
- template<typename _Ret, typename... _Args>
void `std::launder` (_Ret*)(_Args...)=delete
- template<typename _Ret, typename... _Args>
void `std::launder` (_Ret*)(_Args.....)=delete
- template<typename _Tp >
constexpr _Tp * `std::launder` (_Tp * __p) noexcept
- void `std::launder` (const void *)=delete
- void `std::launder` (const volatile void *)=delete
- void `std::launder` (void *)=delete
- void `std::launder` (volatile void *)=delete
- void `operator delete` (void *) noexcept
- void `operator delete` (void *, const std::nothrow_t &) noexcept
- void `operator delete` (void *, std::align_val_t) noexcept
- void `operator delete` (void *, std::align_val_t, const std::nothrow_t &) noexcept
- void `operator delete` (void *, std::size_t) noexcept
- void `operator delete` (void *, std::size_t, std::align_val_t) noexcept
- void `operator delete` (void *, void *) noexcept
- void `operator delete[]` (void *) noexcept
- void `operator delete[]` (void *, const std::nothrow_t &) noexcept
- void `operator delete[]` (void *, std::align_val_t) noexcept
- void `operator delete[]` (void *, std::align_val_t, const std::nothrow_t &) noexcept
- void `operator delete[]` (void *, std::size_t) noexcept
- void `operator delete[]` (void *, std::size_t, std::align_val_t) noexcept
- void `operator delete[]` (void *, void *) noexcept
- void * `operator new` (std::size_t)
- void * `operator new` (std::size_t, const std::nothrow_t &) noexcept
- void * `operator new` (std::size_t, std::align_val_t)

- void * **operator new** (std::size_t, std::align_val_t, const std::nothrow_t &) noexcept
- void * **operator new** (std::size_t, void *__p) noexcept
- void * **operator new[]** (std::size_t)
- void * **operator new[]** (std::size_t, const std::nothrow_t &) noexcept
- void * **operator new[]** (std::size_t, std::align_val_t)
- void * **operator new[]** (std::size_t, std::align_val_t, const std::nothrow_t &) noexcept
- void * **operator new[]** (std::size_t, void *__p) noexcept
- [new_handler](#) **std::set_new_handler** ([new_handler](#)) throw ()

Variables

- constexpr [destroying_delete_t](#) **std::destroying_delete**
- const nothrow_t **std::nothrow**

7.5.1 Detailed Description

This is a Standard C++ Library header.

The header `new` defines several functions to manage dynamic memory and handling memory allocation errors; see https://gcc.gnu.org/onlinedocs/libstdc++/manual/dynamic_memory.html for more.

7.5.2 Function Documentation

operator new()

```
void * operator new (  
    std::size_t )
```

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

7.6 typeinfo File Reference

Classes

- class [std::bad_cast](#)
- class [std::bad_typeid](#)
- class [std::type_info](#)

Namespaces

- namespace [std](#)

Macros

- `#define __GXX_MERGED_TYPEINFO_NAMES`
- `#define __GXX_TYPEINFO_EQUALITY_INLINE`
- `#define _TYPEINFO`

7.6.1 Detailed Description

This is a Standard C++ Library header.

7.7 algorithm File Reference

Macros

- `#define __cpp_lib_parallel_algorithm`
- `#define _GLIBCXX_ALGORITHM`
- `#define _PSTL_ALGORITHM_FORWARD_DECLARED`

7.7.1 Detailed Description

This is a Standard C++ Library header.

7.8 algorithm File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_sample`
- `#define _GLIBCXX_EXPERIMENTAL_ALGORITHM`

Functions

- `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance >
_SampleIterator std::experimental::sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n)`
- `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomNumberGenerator >
_SampleIterator std::experimental::sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomNumberGenerator &&__g)`
- `template<typename _ForwardIterator, typename _Searcher >
_ForwardIterator std::experimental::search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &__searcher)`
- `template<typename _RandomAccessIterator >
void std::experimental::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last)`

7.8.1 Detailed Description

This is a TS C++ Library header.

7.9 algorithm File Reference

Namespaces

- namespace [__gnu_cxx](#)

Macros

- `#define _EXT_ALGORITHM`

Functions

- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`std::pair< _InputIterator, _OutputIterator > gnu_cxx::copy_n (_InputIterator __first, _Size __count, _OutputIterator __result, std::input_iterator_tag)`
- `template<typename _RAIterator, typename _Size, typename _OutputIterator >`
`std::pair< _RAIterator, _OutputIterator > gnu_cxx::copy_n (_RAIterator __first, _Size __count, _OutputIterator __result, std::random_access_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int gnu_cxx::lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `int gnu_cxx::lexicographical_compare_3way (const char *__first1, const char *__last1, const char *__first2, const char *__last2)`
- `int gnu_cxx::lexicographical_compare_3way (const unsigned char *__first1, const unsigned char *__last1, const unsigned char *__first2, const unsigned char *__last2)`
- `template<typename _Tp >`
`const _Tp & gnu_cxx::median (const _Tp &__a, const _Tp &__b, const _Tp &__c)`
- `template<typename _Tp, typename _Compare >`
`const _Tp & gnu_cxx::median (const _Tp &__a, const _Tp &__b, const _Tp &__c, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator, typename _Distance >`
`_RandomAccessIterator gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, _RandomNumberGenerator &__rand, const _Distance __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Distance >`
`_RandomAccessIterator gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, const _Distance __n)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`std::pair< _InputIterator, _OutputIterator > gnu_cxx::copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`
- `template<typename _InputIterator, typename _Tp, typename _Size >`
`void gnu_cxx::count (_InputIterator __first, _InputIterator __last, const _Tp &__value, _Size &__n)`
- `template<typename _InputIterator, typename _Predicate, typename _Size >`
`void gnu_cxx::count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, _Size &__n)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int gnu_cxx::lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`_RandomAccessIterator gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator &__rand)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance >`
`_OutputIterator gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance, typename _RandomNumberGenerator >`
`_OutputIterator gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator &__rand)`

7.9.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.10 algorithm File Reference

Macros

- `#define _PARALLEL_ALGORITHM`

7.10.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.11 any File Reference

Classes

- class [std::any](#)
- class [std::bad_any_cast](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_any`
- `#define _GLIBCXX_ANY`

Functions

- void [std::__throw_bad_any_cast](#) ()
- template<typename _ValueType >
_ValueType [std::any_cast](#) (const [any](#) &__any)
- template<typename _Tp, typename... _Args>
[enable_if_t](#)< is_constructible_v< [any](#), in_place_type_t< _Tp >, _Args... >, [any](#) > [std::make_any](#) (_Args &&... __args)
- template<typename _Tp, typename _Up, typename... _Args>
[enable_if_t](#)< is_constructible_v< [any](#), in_place_type_t< _Tp >, [initializer_list](#)< _Up > &, _Args... >, [any](#) > [std::make_any](#) ([initializer_list](#)< _Up > __il, _Args &&... __args)
- void [std::swap](#) ([any](#) &__x, [any](#) &__y) noexcept
- template<typename _ValueType >
_ValueType [std::any_cast](#) ([any](#) &&__any)
- template<typename _ValueType >
_ValueType [std::any_cast](#) ([any](#) &__any)
- template<typename _ValueType >
_ValueType * [std::any_cast](#) ([any](#) *__any) noexcept
- template<typename _ValueType >
const _ValueType * [std::any_cast](#) (const [any](#) *__any) noexcept

7.11.1 Detailed Description

This is a Standard C++ Library header.

7.12 any File Reference

Classes

- class [std::experimental::fundamentals_v1::any](#)
- class [std::experimental::fundamentals_v1::bad_any_cast](#)

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_any`
- `#define _GLIBCXX_EXPERIMENTAL_ANY`

Functions

- `template<typename _ValueType >`
`_ValueType std::experimental::any_cast (const any &__any)`
- `void std::experimental::swap (any &__x, any &__y) noexcept`

- `template<typename _ValueType , typename enable_if<!is_move_constructible< _ValueType >::value||is_lvalue_reference< _ValueType >::value, bool >::type = true>`
`_ValueType std::experimental::any_cast (any &&__any)`
- `template<typename _ValueType , typename enable_if< is_move_constructible< _ValueType >::value &&!is_lvalue_reference< _ValueType >::value, bool >::type = false>`
`_ValueType std::experimental::any_cast (any &&__any)`
- `template<typename _ValueType >`
`_ValueType std::experimental::any_cast (any &__any)`

- `template<typename _ValueType >`
`_ValueType * std::experimental::any_cast (any *__any) noexcept`
- `template<typename _ValueType >`
`const _ValueType * std::experimental::any_cast (const any *__any) noexcept`

7.12.1 Detailed Description

This is a TS C++ Library header.

7.13 array File Reference

Classes

- struct [std::array<_Tp, _Nm >](#)
- struct [std::tuple_element<_Ind, array<_Tp, _Nm > >](#)
- struct [std::tuple_size< array<_Tp, _Nm > >](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_to_array`
- `#define _GLIBCXX_ARRAY`

Functions

- `template<bool _Move = false, typename _Tp, size_t... _Idx>`
`constexpr array< remove_cv_t< _Tp >, sizeof...(_Idx)> std::__to_array (_Tp(&__a)[sizeof...(_Idx)],`
`index_sequence< _Idx... >)`
- `template<typename _Tp, typename... _Up>`
`std::array (_Tp, _Up...) -> array< enable_if_t<(is_same_v< _Tp, _Up > &&...), _Tp >, 1+sizeof...(_Up)>`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp && std::get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp & std::get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp && std::get (const array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp & std::get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator!= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator< (const array< _Tp, _Nm > &__a, const array< _Tp, _Nm > &__b)`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator<= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator== (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator> (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr bool std::operator>= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>`
`enable_if<!__array_traits< _Tp, _Nm >::is_swappable::value >::type std::swap (array< _Tp, _Nm > &`
`array< _Tp, _Nm > &)=delete`
- `template<typename _Tp, std::size_t _Nm>`
`constexpr enable_if<__array_traits< _Tp, _Nm >::is_swappable::value >::type std::swap (array< _Tp, _Nm`
`> &__one, array< _Tp, _Nm > &__two) noexcept(noexcept(__one.swap(__two)))`
- `template<typename _Tp, size_t _Nm>`
`constexpr array< remove_cv_t< _Tp >, _Nm > std::to_array (_Tp(&&__a)[_Nm]) noexcept(is_nothrow_←`
`move_constructible_v< _Tp >)`
- `template<typename _Tp, size_t _Nm>`
`constexpr array< remove_cv_t< _Tp >, _Nm > std::to_array (_Tp(&__a)[_Nm]) noexcept(is_nothrow_←`
`constructible_v< _Tp, _Tp & >)`

Variables

- `template<typename _Tp, size_t _Nm>`
`constexpr size_t std::tuple_size_v< array< _Tp, _Nm > >`
- `template<typename _Tp, size_t _Nm>`
`constexpr size_t std::tuple_size_v< const array< _Tp, _Nm > >`

7.13.1 Detailed Description

This is a Standard C++ Library header.

7.14 array File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_make_array`
- `#define _GLIBCXX_EXPERIMENTAL_ARRAY`

Functions

- `template<typename _Tp, size_t _Nm, size_t... _Idx>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::__to_array (_Tp(&__a)[_Nm],
index_sequence< _Idx... >)`
- `template<typename _Dest = void, typename... _Types>
constexpr array< typename __make_array_elem< _Dest, _Types... >::type, sizeof...(_Types)> std::experimental::make_array
(_Types &&... __t)`
- `template<typename _Tp, size_t _Nm>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::to_array (_Tp(&__a)[_Nm]) noexcept(is_nothrow_constructible<
remove_cv_t< _Tp >, _Tp &>::value)`

7.14.1 Detailed Description

This is a TS C++ Library header.

7.15 atomic File Reference

Classes

- class [std::atomic](#)< _Tp >
- struct [std::atomic](#)< _Tp * >
- struct [std::atomic](#)< bool >
- struct [std::atomic](#)< char >
- struct [std::atomic](#)< char16_t >
- struct [std::atomic](#)< char32_t >
- struct [std::atomic](#)< int >
- struct [std::atomic](#)< long >
- struct [std::atomic](#)< long long >
- struct [std::atomic](#)< short >
- struct [std::atomic](#)< signed char >
- struct [std::atomic](#)< unsigned char >
- struct [std::atomic](#)< unsigned int >
- struct [std::atomic](#)< unsigned long >
- struct [std::atomic](#)< unsigned long long >
- struct [std::atomic](#)< unsigned short >
- struct [std::atomic](#)< wchar_t >
- struct [std::atomic_ref](#)< _Tp >

Namespaces

- namespace [std](#)

Macros

- #define `__cpp_lib_atomic_float`
- #define `__cpp_lib_atomic_is_always_lock_free`
- #define `__cpp_lib_atomic_ref`
- #define `_GLIBCXX20_INIT(l)`
- #define `_GLIBCXX_ATOMIC`

Typedefs

- typedef `atomic< bool >` `std::atomic_bool`
- typedef `atomic< char >` `std::atomic_char`
- typedef `atomic< char16_t >` `std::atomic_char16_t`
- typedef `atomic< char32_t >` `std::atomic_char32_t`
- typedef `atomic< int >` `std::atomic_int`
- typedef `atomic< int16_t >` `std::atomic_int16_t`
- typedef `atomic< int32_t >` `std::atomic_int32_t`
- typedef `atomic< int64_t >` `std::atomic_int64_t`
- typedef `atomic< int8_t >` `std::atomic_int8_t`
- typedef `atomic< int_fast16_t >` `std::atomic_int_fast16_t`
- typedef `atomic< int_fast32_t >` `std::atomic_int_fast32_t`
- typedef `atomic< int_fast64_t >` `std::atomic_int_fast64_t`
- typedef `atomic< int_fast8_t >` `std::atomic_int_fast8_t`
- typedef `atomic< int_least16_t >` `std::atomic_int_least16_t`
- typedef `atomic< int_least32_t >` `std::atomic_int_least32_t`
- typedef `atomic< int_least64_t >` `std::atomic_int_least64_t`
- typedef `atomic< int_least8_t >` `std::atomic_int_least8_t`
- typedef `atomic< intmax_t >` `std::atomic_intmax_t`
- typedef `atomic< intptr_t >` `std::atomic_intptr_t`
- typedef `atomic< long long >` `std::atomic_llong`
- typedef `atomic< long >` `std::atomic_long`
- typedef `atomic< ptrdiff_t >` `std::atomic_ptrdiff_t`
- typedef `atomic< signed char >` `std::atomic_schar`
- typedef `atomic< short >` `std::atomic_short`
- typedef `atomic< size_t >` `std::atomic_size_t`
- typedef `atomic< unsigned char >` `std::atomic_uchar`
- typedef `atomic< unsigned int >` `std::atomic_uint`
- typedef `atomic< uint16_t >` `std::atomic_uint16_t`
- typedef `atomic< uint32_t >` `std::atomic_uint32_t`
- typedef `atomic< uint64_t >` `std::atomic_uint64_t`
- typedef `atomic< uint8_t >` `std::atomic_uint8_t`
- typedef `atomic< uint_fast16_t >` `std::atomic_uint_fast16_t`
- typedef `atomic< uint_fast32_t >` `std::atomic_uint_fast32_t`
- typedef `atomic< uint_fast64_t >` `std::atomic_uint_fast64_t`
- typedef `atomic< uint_fast8_t >` `std::atomic_uint_fast8_t`
- typedef `atomic< uint_least16_t >` `std::atomic_uint_least16_t`
- typedef `atomic< uint_least32_t >` `std::atomic_uint_least32_t`
- typedef `atomic< uint_least64_t >` `std::atomic_uint_least64_t`
- typedef `atomic< uint_least8_t >` `std::atomic_uint_least8_t`
- typedef `atomic< uintmax_t >` `std::atomic_uintmax_t`
- typedef `atomic< uintptr_t >` `std::atomic_uintptr_t`
- typedef `atomic< unsigned long long >` `std::atomic_ullong`
- typedef `atomic< unsigned long >` `std::atomic_ulong`
- typedef `atomic< unsigned short >` `std::atomic_ushort`
- typedef `atomic< wchar_t >` `std::atomic_wchar_t`

Functions

- `template<typename _ITp >`
`bool std::atomic_compare_exchange_strong (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_strong (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_strong_explicit (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_strong_explicit (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak_explicit (atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak_explicit (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > *__i1, __atomic_val_t< _ITp > __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (volatile atomic< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (volatile atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (volatile atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`

- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (volatile atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (volatile atomic< _ITp > *__a, __atomic_diff_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (__atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (volatile __atomic_base< _ITp > *__a, __atomic_val_t< _ITp > __i, memory_order __m) noexcept`
- `void std::atomic_flag_clear (atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_notify_all (atomic_flag *__a) noexcept`
- `void std::atomic_flag_notify_one (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test (const atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test (const volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_explicit (const atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_explicit (const volatile atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_wait (atomic_flag *__a, bool __old) noexcept`
- `void std::atomic_flag_wait_explicit (atomic_flag *__a, bool __old, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (atomic< _ITp > *__a, __atomic_val_t< _ITp > __i) noexcept`

- `template<typename _ITp >`
`void std::atomic_init (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const volatile atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const volatile atomic<_ITp> *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const atomic<_ITp> *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const volatile atomic<_ITp> *__a, memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::atomic_notify_all (atomic<_Tp> *__a) noexcept`
- `template<typename _Tp >`
`void std::atomic_notify_one (atomic<_Tp> *__a) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (volatile atomic<_ITp> *__a, __atomic_val_t<_ITp> __i, memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::atomic_wait (const atomic<_Tp> *__a, typename std::atomic<_Tp>::value_type __old) noexcept`
- `template<typename _Tp >`
`void std::atomic_wait_explicit (const atomic<_Tp> *__a, typename std::atomic<_Tp>::value_type __old, std::memory_order __m) noexcept`

7.15.1 Detailed Description

This is a Standard C++ Library header.

7.16 `auto_ptr.h` File Reference

Classes

- class [std::auto_ptr<_Tp>](#)
- struct [std::auto_ptr_ref<_Tp1>](#)

Namespaces

- namespace [std](#)

7.16.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.17 backward_warning.h File Reference

7.17.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

7.18 binders.h File Reference

Classes

- class [std::binder1st<_Operation>](#)
- class [std::binder2nd<_Operation>](#)

Namespaces

- namespace [std](#)

Functions

- [template<typename _Operation, typename _Tp>
binder1st<_Operation> std::bind1st](#) (const _Operation &__fn, const _Tp &__x)
- [template<typename _Operation, typename _Tp>
binder2nd<_Operation> std::bind2nd](#) (const _Operation &__fn, const _Tp &__x)

7.18.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.19 hash_fun.h File Reference

Namespaces

- namespace [__gnu_cxx](#)

Functions

- [size_t __gnu_cxx::__stl_hash_string](#) (const char *__s)

7.19.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.20 hash_map File Reference

Classes

- class [__gnu_cxx::hash_map<_Key, _Tp, _HashFn, _EqualKey, _Alloc>](#)
- class [__gnu_cxx::hash_multimap<_Key, _Tp, _HashFn, _EqualKey, _Alloc>](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Macros

- #define **_BACKWARD_HASH_MAP**

Functions

- template<class `_Key` , class `_Tp` , class `_HashFn` , class `_EqKey` , class `_Alloc` >
bool **__gnu_cxx::operator!=** (const [hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm1, const [hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm2)
- template<class `_Key` , class `_Tp` , class `_HF` , class `_EqKey` , class `_Alloc` >
bool **__gnu_cxx::operator!=** (const [hash_multimap](#)< `_Key`, `_Tp`, `_HF`, `_EqKey`, `_Alloc` > &__hm1, const [hash_multimap](#)< `_Key`, `_Tp`, `_HF`, `_EqKey`, `_Alloc` > &__hm2)
- template<class `_Key` , class `_Tp` , class `_HashFn` , class `_EqKey` , class `_Alloc` >
bool **__gnu_cxx::operator==** (const [hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm1, const [hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm2)
- template<class `_Key` , class `_Tp` , class `_HF` , class `_EqKey` , class `_Alloc` >
bool **__gnu_cxx::operator==** (const [hash_multimap](#)< `_Key`, `_Tp`, `_HF`, `_EqKey`, `_Alloc` > &__hm1, const [hash_multimap](#)< `_Key`, `_Tp`, `_HF`, `_EqKey`, `_Alloc` > &__hm2)
- template<class `_Key` , class `_Tp` , class `_HashFn` , class `_EqKey` , class `_Alloc` >
void **__gnu_cxx::swap** ([hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm1, [hash_map](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm2)
- template<class `_Key` , class `_Tp` , class `_HashFn` , class `_EqKey` , class `_Alloc` >
void **__gnu_cxx::swap** ([hash_multimap](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm1, [hash_multimap](#)< `_Key`, `_Tp`, `_HashFn`, `_EqKey`, `_Alloc` > &__hm2)

7.20.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.21 `hash_set` File Reference

Classes

- class [__gnu_cxx::hash_multiset](#)< `_Value`, `_HashFcn`, `_EqualKey`, `_Alloc` >
- class [__gnu_cxx::hash_set](#)< `_Value`, `_HashFcn`, `_EqualKey`, `_Alloc` >

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Macros

- #define **_BACKWARD_HASH_SET**

Functions

- template<class `_Val` , class `_HashFcn` , class `_EqualKey` , class `_Alloc` >
bool **__gnu_cxx::operator!=** (const [hash_multiset](#)< `_Val`, `_HashFcn`, `_EqualKey`, `_Alloc` > &__hs1, const [hash_multiset](#)< `_Val`, `_HashFcn`, `_EqualKey`, `_Alloc` > &__hs2)
- template<class `_Value` , class `_HashFcn` , class `_EqualKey` , class `_Alloc` >
bool **__gnu_cxx::operator!=** (const [hash_set](#)< `_Value`, `_HashFcn`, `_EqualKey`, `_Alloc` > &__hs1, const [hash_set](#)< `_Value`, `_HashFcn`, `_EqualKey`, `_Alloc` > &__hs2)

- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`bool __gnu_cxx::operator== (const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc >`
`bool __gnu_cxx::operator== (const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void __gnu_cxx::swap (hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void __gnu_cxx::swap (hash_set< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, hash_set< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`

7.21.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.22 **strstream** File Reference

Namespaces

- namespace [std](#)

7.22.1 Detailed Description

This is a Standard C++ Library header.

7.23 **barrier** File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_barrier`
- `#define _GLIBCXX_BARRIER`

Enumerations

- enum class `__barrier_phase_t`: unsigned char

7.23.1 Detailed Description

This is a Standard C++ Library header.

7.24 **bit** File Reference

Namespaces

- namespace [std](#)

Macros

- #define `__cpp_lib_bit_cast`
- #define `__cpp_lib_bitops`
- #define `__cpp_lib_endian`
- #define `__cpp_lib_int_pow2`
- #define `_GLIBCXX_BIT`

Enumerations

- enum class `std::endian` { `little` , `big` , `native` }

Functions

- template<typename `_To` , typename `_From` >
requires (sizeof(`_To`) == sizeof(`_From`)) && `__is_trivially_copyable`(`_To`) && `__is_trivially_copyable`(`_From`)
constexpr `_To` `std::bit_cast` (const `_From` &`__from`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp` > `std::bit_ceil` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp` > `std::bit_floor` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp` > `std::bit_width` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, int > `std::countl_one` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, int > `std::countl_zero` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, int > `std::countr_one` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, int > `std::countr_zero` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, bool > `std::has_single_bit` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp`, int > `std::popcount` (`_Tp` `__x`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp` > `std::rotr` (`_Tp` `__x`, int `__s`) noexcept
- template<typename `_Tp` >
constexpr `_If_is_unsigned_integer`< `_Tp` > `std::rotr` (`_Tp` `__x`, int `__s`) noexcept

7.24.1 Detailed Description

This is a Standard C++ Library header.

7.25 `algorithmfwd.h` File Reference

Namespaces

- namespace `std`

Macros

- #define `__cpp_lib_constexpr_algorithms`

Functions

- `template<typename _Filter >`
`constexpr _Filter std::adjacent_find (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`constexpr _Filter std::adjacent_find (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool std::all_of (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool std::any_of (_Iter, _Iter, _Predicate)`
- `template<typename _Filter, typename _Tp >`
`constexpr bool std::binary_search (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr bool std::binary_search (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _Tp >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi, _Compare __comp)`
- `template<typename _Iter, typename _OIter >`
`constexpr _OIter std::copy (_Iter, _Iter, _OIter)`
- `template<typename _BIter1, typename _BIter2 >`
`constexpr _BIter2 std::copy_backward (_BIter1, _BIter1, _BIter2)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`constexpr _OIter std::copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Iter, typename _Size, typename _OIter >`
`constexpr _OIter std::copy_n (_Iter, _Size, _OIter)`
- `template<typename _Iter, typename _Tp >`
`constexpr iterator_traits< _Iter >::difference_type std::count (_Iter, _Iter, const _Tp &)`
- `template<typename _Iter, typename _Predicate >`
`constexpr iterator_traits< _Iter >::difference_type std::count_if (_Iter, _Iter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`constexpr bool std::equal (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::equal (_Iter1, _Iter1, _Iter2)`
- `template<typename _Filter, typename _Tp >`
`constexpr pair< _Filter, _Filter > std::equal_range (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr pair< _Filter, _Filter > std::equal_range (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _Filter, typename _Tp >`
`constexpr void std::fill (_Filter, _Filter, const _Tp &)`
- `template<typename _OIter, typename _Size, typename _Tp >`
`constexpr _OIter std::fill_n (_OIter, _Size, const _Tp &)`
- `template<typename _Iter, typename _Tp >`
`constexpr _Iter std::find (_Iter, _Iter, const _Tp &)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 std::find_end (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 std::find_end (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 std::find_first_of (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 std::find_first_of (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`

- `template<typename _Iter, typename _Predicate >`
`constexpr _Iter std::find_if (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Predicate >`
`constexpr _Iter std::find_if_not (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Funct >`
`constexpr _Funct std::for_each (_Iter, _Iter, _Funct)`
- `template<typename _Filter, typename _Generator >`
`constexpr void std::generate (_Filter, _Filter, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`constexpr _OIter std::generate_n (_OIter, _Size, _Generator)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::includes (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`constexpr bool std::includes (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _BIter >`
`void std::inplace_merge (_BIter, _BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`void std::inplace_merge (_BIter, _BIter, _BIter, _Compare)`
- `template<typename _RAIter >`
`constexpr bool std::is_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr bool std::is_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`constexpr _RAIter std::is_heap_until (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr _RAIter std::is_heap_until (_RAIter, _RAIter, _Compare)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool std::is_partitioned (_Iter, _Iter, _Predicate)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr bool std::is_permutation (_Filter1, _Filter1, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_Filter1, _Filter1, _Filter2, _BinaryPredicate)`
- `template<typename _Filter >`
`constexpr bool std::is_sorted (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr bool std::is_sorted (_Filter, _Filter, _Compare)`
- `template<typename _Filter >`
`constexpr _Filter std::is_sorted_until (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr _Filter std::is_sorted_until (_Filter, _Filter, _Compare)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr void std::iter_swap (_Filter1, _Filter2)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`constexpr bool std::lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _Filter, typename _Tp >`
`constexpr _Filter std::lower_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr _Filter std::lower_bound (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _RAIter >`
`constexpr void std::make_heap (_RAIter, _RAIter)`

- `template<typename _RAIter, typename _Compare >`
`constexpr void std::make_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _Tp >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr _Tp std::max (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr _Tp std::max (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`constexpr _Filter std::max_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr _Filter std::max_element (_Filter, _Filter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`constexpr _OIter std::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`constexpr _OIter std::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _Tp >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr _Tp std::min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr _Tp std::min (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`constexpr _Filter std::min_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr _Filter std::min_element (_Filter, _Filter, _Compare)`
- `template<typename _Tp >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`constexpr pair< _Filter, _Filter > std::minmax_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`constexpr pair< _Filter, _Filter > std::minmax_element (_Filter, _Filter, _Compare)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr pair< _Iter1, _Iter2 > std::mismatch (_Iter1, _Iter1, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`constexpr pair< _Iter1, _Iter2 > std::mismatch (_Iter1, _Iter1, _Iter2, _BinaryPredicate)`
- `template<typename _BIter >`
`constexpr bool std::next_permutation (_BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`constexpr bool std::next_permutation (_BIter, _BIter, _Compare)`
- `template<typename _Iter, typename _Predicate >`
`constexpr bool std::none_of (_Iter, _Iter, _Predicate)`

- `template<typename _RAIter >`
`constexpr void std::nth_element (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void std::nth_element (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`constexpr void std::partial_sort (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void std::partial_sort (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _Iter, typename _RAIter >`
`constexpr _RAIter std::partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter)`
- `template<typename _Iter, typename _RAIter, typename _Compare >`
`constexpr _RAIter std::partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter, _Compare)`
- `template<typename _BIter, typename _Predicate >`
`constexpr _BIter std::partition (_BIter, _BIter, _Predicate)`
- `template<typename _Iter, typename _OIter1, typename _OIter2, typename _Predicate >`
`constexpr pair< _OIter1, _OIter2 > std::partition_copy (_Iter, _Iter, _OIter1, _OIter2, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`constexpr _Filter std::partition_point (_Filter, _Filter, _Predicate)`
- `template<typename _RAIter >`
`constexpr void std::pop_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void std::pop_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _BIter >`
`constexpr bool std::prev_permutation (_BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`constexpr bool std::prev_permutation (_BIter, _BIter, _Compare)`
- `template<typename _RAIter >`
`constexpr void std::push_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`constexpr void std::push_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`void std::random_shuffle (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Generator >`
`void std::random_shuffle (_RAIter, _RAIter, _Generator &&)`
- `template<typename _Filter, typename _Tp >`
`constexpr _Filter std::remove (_Filter, _Filter, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`constexpr _OIter std::remove_copy (_Iter, _Iter, _OIter, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`constexpr _OIter std::remove_copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`constexpr _Filter std::remove_if (_Filter, _Filter, _Predicate)`
- `template<typename _Filter, typename _Tp >`
`constexpr void std::replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`constexpr _OIter std::replace_copy (_Iter, _Iter, _OIter, const _Tp &, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Predicate, typename _Tp >`
`constexpr _OIter std::replace_copy_if (_Iter, _Iter, _OIter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`constexpr void std::replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _BIter >`
`constexpr void std::reverse (_BIter, _BIter)`

- `template<typename _Blter, typename _Olter >`
`constexpr _Olter std::reverse_copy (_Blter, _Blter, _Olter)`
- `template<typename _Filter >`
`constexpr _Filter std::rotate (_Filter, _Filter, _Filter)`
- `template<typename _Filter, typename _Olter >`
`constexpr _Olter std::rotate_copy (_Filter, _Filter, _Filter, _Olter)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter1 std::search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`constexpr _Filter1 std::search (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _Filter, typename _Size, typename _Tp >`
`constexpr _Filter std::search_n (_Filter, _Filter, _Size, const _Tp &)`
- `template<typename _Filter, typename _Size, typename _Tp, typename _BinaryPredicate >`
`constexpr _Filter std::search_n (_Filter, _Filter, _Size, const _Tp &, _BinaryPredicate)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter >`
`constexpr _Olter std::set_difference (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter, typename _Compare >`
`constexpr _Olter std::set_difference (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter, _Compare)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter >`
`constexpr _Olter std::set_intersection (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter, typename _Compare >`
`constexpr _Olter std::set_intersection (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter, _Compare)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter >`
`constexpr _Olter std::set_symmetric_difference (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter, typename _Compare >`
`constexpr _Olter std::set_symmetric_difference (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter, _Compare)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter >`
`constexpr _Olter std::set_union (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter, typename _Compare >`
`constexpr _Olter std::set_union (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter, _Compare)`
- `template<typename _RAlter, typename _UGenerator >`
`void std::shuffle (_RAlter, _RAlter, _UGenerator &&)`
- `template<typename _RAlter >`
`constexpr void std::sort (_RAlter, _RAlter)`
- `template<typename _RAlter, typename _Compare >`
`constexpr void std::sort (_RAlter, _RAlter, _Compare)`
- `template<typename _RAlter >`
`constexpr void std::sort_heap (_RAlter, _RAlter)`
- `template<typename _RAlter, typename _Compare >`
`constexpr void std::sort_heap (_RAlter, _RAlter, _Compare)`
- `template<typename _Blter, typename _Predicate >`
`_Blter std::stable_partition (_Blter, _Blter, _Predicate)`
- `template<typename _RAlter >`
`void std::stable_sort (_RAlter, _RAlter)`
- `template<typename _RAlter, typename _Compare >`
`void std::stable_sort (_RAlter, _RAlter, _Compare)`
- `template<typename _Filter1, typename _Filter2 >`
`constexpr _Filter2 std::swap_ranges (_Filter1, _Filter1, _Filter2)`
- `template<typename _Ilter, typename _Olter, typename _UnaryOperation >`
`constexpr _Olter std::transform (_Ilter, _Ilter, _Olter, _UnaryOperation)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter, typename _BinaryOperation >`
`constexpr _Olter std::transform (_Ilter1, _Ilter1, _Ilter2, _Olter, _BinaryOperation)`

- `template<typename _Filter >`
`constexpr _Filter std::unique (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`constexpr _Filter std::unique (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _Iter, typename _OIter >`
`constexpr _OIter std::unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _BinaryPredicate >`
`constexpr _OIter std::unique_copy (_Iter, _Iter, _OIter, _BinaryPredicate)`
- `template<typename _Filter, typename _Tp >`
`constexpr _Filter std::upper_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`constexpr _Filter std::upper_bound (_Filter, _Filter, const _Tp &, _Compare)`

7.25.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

7.26 `algorithmfwd.h` File Reference

Namespaces

- namespace `std`
- namespace `std::__parallel`

Functions

- `template<typename _Filter, typename _BiPredicate, typename _IterTag >`
`_Filter std::__parallel::__adjacent_find_switch (_Filter, _Filter, _BiPredicate, _IterTag)`
- `template<typename _Filter, typename _IterTag >`
`_Filter std::__parallel::__adjacent_find_switch (_Filter, _Filter, _IterTag)`
- `template<typename _RAIter >`
`_RAIter std::__parallel::__adjacent_find_switch (_RAIter __begin, _RAIter __end, random_access_iterator_tag)`
- `template<typename _RAIter, typename _BiPredicate >`
`_RAIter std::__parallel::__adjacent_find_switch (_RAIter, _RAIter, _BiPredicate, random_access_iterator_tag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`iterator_traits< _Iter >::difference_type std::__parallel::__count_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename _RAIter, typename _Predicate >`
`iterator_traits< _RAIter >::difference_type std::__parallel::__count_if_switch (_RAIter __begin, _RAIter __↵
end, _Predicate __pred, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`iterator_traits< _Iter >::difference_type std::__parallel::__count_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`iterator_traits< _RAIter >::difference_type std::__parallel::__count_switch (_RAIter __begin, _RAIter __end,
const _Tp & __value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Iter std::__parallel::__find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _Filter, typename _IterTag1, typename _IterTag2 >`
`_Iter std::__parallel::__find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _Filter, typename _BiPredicate, typename _IterTag >`
`_RAIter std::__parallel::__find_first_of_switch (_RAIter, _RAIter, _Filter, _Filter, _BiPredicate, random_access_iterator_tag,
_IterTag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`_Iter std::__parallel::__find_if_switch (_Iter, _Iter, _Predicate, _IterTag)`

- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::parallel::find_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`_Iter std::parallel::find_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`_RAIter std::parallel::find_switch (_RAIter __begin, _RAIter __end, const _Tp & __val, random_access_iterator_tag)`
- `template<typename _Iter, typename _Function, typename _IterTag >`
`_Function std::parallel::for_each_switch (_Iter, _Iter, _Function, _IterTag)`
- `template<typename _RAIter, typename _Function >`
`_Function std::parallel::for_each_switch (_RAIter __begin, _RAIter __end, _Function __f, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _OIter, typename _Size, typename _Generator, typename _IterTag >`
`_OIter std::parallel::generate_n_switch (_OIter, _Size, _Generator, _IterTag)`
- `template<typename _RAIter, typename _Size, typename _Generator >`
`_RAIter std::parallel::generate_n_switch (_RAIter __begin, _Size __n, _Generator __gen, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Generator, typename _IterTag >`
`void std::parallel::generate_switch (_Filter, _Filter, _Generator, _IterTag)`
- `template<typename _RAIter, typename _Generator >`
`void std::parallel::generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`bool std::parallel::lexicographical_compare_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter std::parallel::max_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter std::parallel::merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::parallel::merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter std::parallel::min_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch_switch (_Iter1, _Iter1, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _IterTag >`
`_Filter std::parallel::partition_switch (_Filter, _Filter, _Predicate, _IterTag)`

- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::parallel::partition_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp, typename _IterTag >`
`void std::parallel::replace_if_switch (_Filter, _Filter, _Predicate, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, const _Tp &↵
__new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Tp, typename _IterTag >`
`void std::parallel::replace_switch (_Filter, _Filter, const _Tp &, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`void std::parallel::replace_switch (_RAIter __begin, _RAIter __end, const _Tp & __old_value, const _Tp
& __new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate, typename _IterTag >`
`_Filter std::parallel::search_n_switch (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, _IterTag)`
- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_RAIter std::parallel::search_n_switch (_RAIter, _RAIter, _Integer, const _Tp &, _BiPredicate,
random_access_iterator_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Filter1 std::parallel::search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Filter1, typename _Filter2, typename _IterTag1, typename _IterTag2 >`
`_Filter1 std::parallel::search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2 >`
`_RAIter1 std::parallel::search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2
__end2, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _BiPredicate >`
`_RAIter1 std::parallel::search_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter2, _BiPredicate,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >`
`_OIter std::parallel::set_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1,
_IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter std::parallel::set_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2
__begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >`
`_OIter std::parallel::set_intersection_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _Iter↵
Tag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter std::parallel::set_intersection_switch (_RAIter1 __begin1, _RAIter1 __end1, ↵
RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2 ,
typename _IterTag3 >`
`_OIter std::parallel::set_symmetric_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, ↵
Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >`
`_Output_RAIter std::parallel::set_symmetric_difference_switch (_RAIter1 __begin1, _RAIter1 __end1,
_RAlter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >`

- `_OIter std::parallel::set_union_switch (_IOIter1, _IOIter1, _IOIter2, _IOIter2, _OIter, _Predicate, _IOIterTag1, _IOIterTag2, _IOIterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAIter, typename _Predicate >
_Output_RAIter std::parallel::set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAIter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IOIter, typename _OIter, typename _UnaryOperation, typename _IOIterTag1, typename _IOIterTag2 >
_OIter std::parallel::transform1_switch (_IOIter, _IOIter, _OIter, _UnaryOperation, _IOIterTag1, _IOIterTag2)`
- `template<typename _RAIOIter, typename _RAOIter, typename _UnaryOperation >
_RAOIter std::parallel::transform1_switch (_RAIOIter, _RAIOIter, _RAOIter, _UnaryOperation, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism= __gnu_parallel::parallel_balanced)`
- `template<typename _IOIter1, typename _IOIter2, typename _OIter, typename _BiOperation, typename _Tag1, typename _Tag2, typename _Tag3 >
_OIter std::parallel::transform2_switch (_IOIter1, _IOIter1, _IOIter2, _OIter, _BiOperation, _Tag1, _Tag2, _IOIterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BiOperation >
_RAIter3 std::parallel::transform2_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter3, _BiOperation, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism= __gnu_parallel::parallel_balanced)`
- `template<typename _IOIter, typename _OIter, typename _Predicate, typename _IOIterTag1, typename _IOIterTag2 >
_OIter std::parallel::unique_copy_switch (_IOIter, _IOIter, _OIter, _Predicate, _IOIterTag1, _IOIterTag2)`
- `template<typename _RAIter, typename _RandomAccess_OIter, typename _Predicate >
_RandomAccess_OIter std::parallel::unique_copy_switch (_RAIter, _RAIter, _RandomAccess_OIter, _IOIterTag1, _IOIterTag2, _Predicate, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter >
_Filter std::parallel::adjacent_find (_Filter, _Filter)`
- `template<typename _Filter >
_Filter std::parallel::adjacent_find (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _BiPredicate >
_Filter std::parallel::adjacent_find (_Filter, _Filter, _BiPredicate)`
- `template<typename _Filter, typename _BiPredicate >
_Filter std::parallel::adjacent_find (_Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _IOIter, typename _Tp >
iterator_traits< _IOIter >::difference_type std::parallel::count (_IOIter __begin, _IOIter __end, const _Tp & __value)`
- `template<typename _IOIter, typename _Tp >
iterator_traits< _IOIter >::difference_type std::parallel::count (_IOIter __begin, _IOIter __end, const _Tp & __value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IOIter, typename _Tp >
iterator_traits< _IOIter >::difference_type std::parallel::count (_IOIter __begin, _IOIter __end, const _Tp & __value, __gnu_parallel::sequential_tag)`
- `template<typename _IOIter, typename _Predicate >
iterator_traits< _IOIter >::difference_type std::parallel::count_if (_IOIter __begin, _IOIter __end, _Predicate __pred)`
- `template<typename _IOIter, typename _Predicate >
iterator_traits< _IOIter >::difference_type std::parallel::count_if (_IOIter __begin, _IOIter __end, _Predicate __pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IOIter, typename _Predicate >
iterator_traits< _IOIter >::difference_type std::parallel::count_if (_IOIter __begin, _IOIter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _IOIter1, typename _IOIter2 >
constexpr bool std::parallel::equal (_IOIter1 __begin1, _IOIter1 __end1, _IOIter2 __begin2)`

- `template<typename _Iter1, typename _Iter2 >`
`bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`constexpr bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Iter std::__parallel::find (_Iter __begin, _Iter __end, const _Tp &__val)`
- `template<typename _Iter, typename _Tp >`
`_Iter std::__parallel::find (_Iter __begin, _Iter __end, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter >`
`_Iter std::__parallel::find_first_of (_Iter, _Iter, _Filter, _Filter)`
- `template<typename _Iter, typename _Filter >`
`_Iter std::__parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter std::__parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter std::__parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::__parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::__parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Function >`
`_Function std::__parallel::for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Function >`
`_Function std::__parallel::for_each (_Iter, _Iter, _Function)`
- `template<typename _Iterator, typename _Function >`
`_Function std::__parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::Parallelism_parallelism_tag)`
- `template<typename _Filter, typename _Generator >`
`void std::__parallel::generate (_Filter, _Filter, _Generator)`
- `template<typename _Filter, typename _Generator >`
`void std::__parallel::generate (_Filter, _Filter, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Generator >`
`void std::__parallel::generate (_Filter, _Filter, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::__parallel::generate_n (_OIter, _Size, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::__parallel::generate_n (_OIter, _Size, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::__parallel::generate_n (_OIter, _Size, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`constexpr bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred)`

- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter std::__parallel::max_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter std::__parallel::max_element (_Filter, _Filter, __gnu_parallel::Parallelism)`
- `template<typename _Filter >`
`_Filter std::__parallel::max_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::max_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::max_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::max_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::__parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::__parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter std::__parallel::min_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter std::__parallel::min_element (_Filter, _Filter, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter >`
`_Filter std::__parallel::min_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::min_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::min_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::__parallel::min_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`

- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp,`
`__gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::__parallel::partition (_Filter, _Filter, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::__parallel::partition (_Filter, _Filter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &__rand,`
`__gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::__parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::__parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::Parallelism)`

- `template<typename _Iter, typename _OIter, typename _UnaryOperation >
_OIter std::parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >
_OIter std::parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >
_OIter std::parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >
_OIter std::parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >
_OIter std::parallel::unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter >
_OIter std::parallel::unique_copy (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _Predicate >
_OIter std::parallel::unique_copy (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Iter, typename _OIter, typename _Predicate >
_OIter std::parallel::unique_copy (_Iter, _Iter, _OIter, _Predicate, __gnu_parallel::sequential_tag)`

7.26.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

7.27 align.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_assume_aligned`

Functions

- `void * std::align (size_t __align, size_t __size, void *&__ptr, size_t &__space) noexcept`
- `template<size_t _Align, class _Tp >
constexpr _Tp * std::assume_aligned (_Tp * __ptr) noexcept`

7.27.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.28 alloc_traits.h File Reference

Classes

- struct [std::allocator_traits< _Alloc >](#)
- struct [std::allocator_traits< allocator< _Tp > >](#)
- struct [std::allocator_traits< allocator< void > >](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_allocator_traits_is_always_equal`
- `#define __cpp_lib_constexpr_dynamic_alloc`

Typedefs

- `template<typename _Alloc >`
`using std::__RequireAllocator = typename enable_if< __is_allocator< _Alloc >::value, _Alloc >::type`
- `template<typename _Alloc >`
`using std::__RequireNotAllocator = typename enable_if<!
__is_allocator< _Alloc >::value, _Alloc >::type`

Functions

- `template<typename _Alloc >`
`constexpr void std::__alloc_on_copy (_Alloc &__one, const _Alloc &__two)`
- `template<typename _Alloc >`
`constexpr _Alloc std::__alloc_on_copy (const _Alloc &__a)`
- `template<typename _Alloc >`
`constexpr void std::__alloc_on_move (_Alloc &__one, _Alloc &__two)`
- `template<typename _Alloc >`
`constexpr void std::__alloc_on_swap (_Alloc &__one, _Alloc &__two)`
- `template<typename _ForwardIterator, typename _Allocator >`
`constexpr void std::__Destroy (_ForwardIterator __first, _ForwardIterator __last, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::__Destroy (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp > &)`

7.28.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.29 `alloc_traits.h` File Reference

Classes

- struct [__gnu_cxx::__alloc_traits](#)< _Alloc, typename >

Namespaces

- namespace [__gnu_cxx](#)

7.29.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.30 `allocated_ptr.h` File Reference

Namespaces

- namespace [std](#)

7.30.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.31 allocator.h File Reference

Classes

- class [std::allocator< _Tp >](#)
- class [std::allocator< void >](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_incomplete_container_elements`

Functions

- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator== (const allocator< _T1 > &, const allocator< _T2 > &) noexcept`

7.31.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.32 atomic_base.h File Reference

Classes

- struct [std::__atomic_base< _ITp >](#)
- struct [std::__atomic_base< _PTp * >](#)
- struct [std::__atomic_flag_base](#)
- struct [std::atomic_flag](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_atomic_flag_test`
- `#define __cpp_lib_atomic_value_initialization`
- `#define _GLIBCXX20_INIT(l)`
- `#define ATOMIC_FLAG_INIT`
- `#define ATOMIC_VAR_INIT(_VI)`

Typedefs

- `typedef unsigned char std::__atomic_flag_data_type`
- `template<typename _Tp >`
`using std::__atomic_impl::Diff = __conditional_t< is_pointer_v< _Tp >, ptrdiff_t, _Val< _Tp > >`
- `template<typename _Tp >`
`using std::__atomic_impl::Val = remove_volatile_t< _Tp >`

Enumerations

- enum `__memory_order_modifier` { `__memory_order_mask` , `__memory_order_modifier_mask` , `__memory_order_hle_acquire` , `__memory_order_hle_release` }
- enum class `std::memory_order` : int { `relaxed` , `consume` , `acquire` , `release` , `acq_rel` , `seq_cst` }

Functions

- template<typename `_Tp` >
`_Tp std::__atomic_impl::__add_fetch` (`_Tp *``__ptr`, `_Diff`< `_Tp` > `__i`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__add_fetch_flt` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__and_fetch` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`) noexcept
- constexpr `memory_order std::__cmpexch_failure_order` (`memory_order` `__m`) noexcept
- constexpr `memory_order std::__cmpexch_failure_order2` (`memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__fetch_add_flt` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__fetch_sub_flt` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- constexpr bool `std::__is_valid_cmpexch_failure_order` (`memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__or_fetch` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__sub_fetch` (`_Tp *``__ptr`, `_Diff`< `_Tp` > `__i`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__sub_fetch_flt` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::__xor_fetch` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`) noexcept
- void `std::atomic_signal_fence` (`memory_order` `__m`) noexcept
- void `std::atomic_thread_fence` (`memory_order` `__m`) noexcept
- template<typename `_Tp` >
bool `std::__atomic_impl::compare_exchange_strong` (`_Tp *``__ptr`, `_Val`< `_Tp` > &`__expected`, `_Val`< `_Tp` > `__desired`, `memory_order` `__success`, `memory_order` `__failure`) noexcept
- template<typename `_Tp` >
bool `std::__atomic_impl::compare_exchange_weak` (`_Tp *``__ptr`, `_Val`< `_Tp` > &`__expected`, `_Val`< `_Tp` > `__desired`, `memory_order` `__success`, `memory_order` `__failure`) noexcept
- template<typename `_Tp` >
`_Val`< `_Tp` > `std::__atomic_impl::exchange` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__desired`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::fetch_add` (`_Tp *``__ptr`, `_Diff`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::fetch_and` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::fetch_or` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::fetch_sub` (`_Tp *``__ptr`, `_Diff`< `_Tp` > `__i`, `memory_order` `__m`) noexcept
- template<typename `_Tp` >
`_Tp std::__atomic_impl::fetch_xor` (`_Tp *``__ptr`, `_Val`< `_Tp` > `__i`, `memory_order` `__m`) noexcept

- `template<size_t _Size, size_t _Align>`
`bool std::__atomic_impl::is_lock_free () noexcept`
- `template<typename _Tp >`
`_Tp std::kill_dependency (_Tp __y) noexcept`
- `template<typename _Tp >`
`_Val< _Tp > std::__atomic_impl::load (const _Tp *__ptr, memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::__atomic_impl::notify_all (const _Tp *__ptr) noexcept`
- `template<typename _Tp >`
`void std::__atomic_impl::notify_one (const _Tp *__ptr) noexcept`
- `constexpr memory_order std::operator& (memory_order __m, __memory_order_modifier __mod)`
- `constexpr memory_order std::operator| (memory_order __m, __memory_order_modifier __mod)`
- `template<typename _Tp >`
`void std::__atomic_impl::store (_Tp *__ptr, _Val< _Tp > __t, memory_order __m) noexcept`
- `template<typename _Tp >`
`void std::__atomic_impl::wait (const _Tp *__ptr, _Val< _Tp > __old, memory_order __m=memory_order_↔
seq_cst) noexcept`

Variables

- `constexpr memory_order std::memory_order_acq_rel`
- `constexpr memory_order std::memory_order_acquire`
- `constexpr memory_order std::memory_order_consume`
- `constexpr memory_order std::memory_order_relaxed`
- `constexpr memory_order std::memory_order_release`
- `constexpr memory_order std::memory_order_seq_cst`

7.32.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

7.33 atomic_futex.h File Reference

Namespaces

- namespace `std`

7.33.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.34 atomic_lockfree_defines.h File Reference

Macros

- `#define ATOMIC_BOOL_LOCK_FREE`
- `#define ATOMIC_CHAR16_T_LOCK_FREE`
- `#define ATOMIC_CHAR32_T_LOCK_FREE`
- `#define ATOMIC_CHAR_LOCK_FREE`
- `#define ATOMIC_INT_LOCK_FREE`
- `#define ATOMIC_LLONG_LOCK_FREE`
- `#define ATOMIC_LONG_LOCK_FREE`
- `#define ATOMIC_POINTER_LOCK_FREE`
- `#define ATOMIC_SHORT_LOCK_FREE`
- `#define ATOMIC_WCHAR_T_LOCK_FREE`

7.34.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

7.35 atomic_timed_wait.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Typedefs

- using [std::__detail::__bare_timed_wait](#) = [__timed_waiter](#)< [std::false_type](#) >
- using [std::__detail::__enters_timed_wait](#) = [__timed_waiter](#)< [std::true_type](#) >
- using [std::__detail::__wait_clock_t](#) = [chrono::steady_clock](#)

Functions

- [template](#)<typename [_Tp](#), typename [_Pred](#), typename [_Rep](#), typename [_Period](#) >
bool [std::__atomic_wait_address_for](#) (const [_Tp](#) *[__addr](#), [_Pred](#) [__pred](#), const [chrono::duration](#)< [_Rep](#), [_Period](#) > &[__rtime](#)) noexcept
- [template](#)<typename [_Pred](#), typename [_Rep](#), typename [_Period](#) >
bool [std::__atomic_wait_address_for_bare](#) (const [std::__detail::__platform_wait_t](#) *[__addr](#), [_Pred](#) [__pred](#), const [chrono::duration](#)< [_Rep](#), [_Period](#) > &[__rtime](#)) noexcept
- [template](#)<typename [_Tp](#), typename [_ValFn](#), typename [_Rep](#), typename [_Period](#) >
bool [std::__atomic_wait_address_for_v](#) (const [_Tp](#) *[__addr](#), [_Tp](#) &&[__old](#), [_ValFn](#) &&[__vfn](#), const [chrono::duration](#)< [_Rep](#), [_Period](#) > &[__rtime](#)) noexcept
- [template](#)<typename [_Tp](#), typename [_Pred](#), typename [_Clock](#), typename [_Dur](#) >
bool [std::__atomic_wait_address_until](#) (const [_Tp](#) *[__addr](#), [_Pred](#) [__pred](#), const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#)) noexcept
- [template](#)<typename [_Pred](#), typename [_Clock](#), typename [_Dur](#) >
bool [std::__atomic_wait_address_until_bare](#) (const [std::__detail::__platform_wait_t](#) *[__addr](#), [_Pred](#) [__pred](#), const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#)) noexcept
- [template](#)<typename [_Tp](#), typename [_ValFn](#), typename [_Clock](#), typename [_Dur](#) >
bool [std::__atomic_wait_address_until_v](#) (const [_Tp](#) *[__addr](#), [_Tp](#) &&[__old](#), [_ValFn](#) &&[__vfn](#), const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#)) noexcept
- [template](#)<typename [_Clock](#), typename [_Dur](#) >
bool [std::__detail::__cond_wait_until](#) ([__condvar](#) &[__cv](#), [mutex](#) &[__mx](#), const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#))
- [template](#)<typename [_Clock](#), typename [_Dur](#) >
bool [std::__detail::__cond_wait_until_impl](#) ([__condvar](#) &[__cv](#), [mutex](#) &[__mx](#), const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#))
- [template](#)<typename [_Dur](#) >
[__wait_clock_t::time_point](#) [std::__detail::__to_wait_clock](#) (const [chrono::time_point](#)< [__wait_clock_t](#), [_Dur](#) > &[__atime](#)) noexcept
- [template](#)<typename [_Clock](#), typename [_Dur](#) >
[__wait_clock_t::time_point](#) [std::__detail::__to_wait_clock](#) (const [chrono::time_point](#)< [_Clock](#), [_Dur](#) > &[__atime](#)) noexcept

7.35.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

7.36 atomic_wait.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_atomic_wait`

Typedefs

- using `std::__detail::__bare_wait` = `__waiter< std::false_type >`
- using `std::__detail::__enters_wait` = `__waiter< std::true_type >`
- using `std::__detail::__platform_wait_t` = `uint64_t`

Functions

- `template<typename _Tp >`
`bool std::__detail::__atomic_compare (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp >`
`void std::__atomic_notify_address (const _Tp *__addr, bool __all) noexcept`
- `void std::__atomic_notify_address_bare (const __detail::__platform_wait_t *__addr, bool __all) noexcept`
- `template<typename _Pred, typename _Spin = __default_spin_policy>`
`bool std::__detail::__atomic_spin (_Pred &__pred, _Spin __spin=_Spin{ }) noexcept`
- `template<typename _Tp, typename _Pred >`
`void std::__atomic_wait_address (const _Tp *__addr, _Pred __pred) noexcept`
- `template<typename _Pred >`
`void std::__atomic_wait_address_bare (const __detail::__platform_wait_t *__addr, _Pred __pred) noexcept`
- `template<typename _Tp, typename _ValFn >`
`void std::__atomic_wait_address_v (const _Tp *__addr, _Tp __old, _ValFn __vfn) noexcept`
- `void std::__detail::__thread_relax () noexcept`
- `void std::__detail::__thread_yield () noexcept`

Variables

- `constexpr auto std::__detail::__atomic_spin_count`
- `constexpr auto std::__detail::__atomic_spin_count_relax`
- `constexpr size_t std::__detail::__platform_wait_alignment`
- `template<typename _Tp >`
`constexpr bool std::__platform_wait_uses_type`

7.36.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

7.37 basic_ios.h File Reference

Classes

- class [std::basic_ios< _CharT, _Traits >](#)

Namespaces

- namespace [std](#)

Functions

- `template<typename _Facet >`
`const _Facet & std::__check_facet (const _Facet *__f)`

7.37.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

7.38 `basic_ios.tcc` File Reference**Namespaces**

- namespace [std](#)

Macros

- `#define _BASIC_IOS_TCC`

7.38.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

7.39 `basic_string.h` File Reference**Classes**

- struct [std::hash< string >](#)
- struct [std::hash< u16string >](#)
- struct [std::hash< u32string >](#)
- struct [std::hash< wstring >](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)
- namespace [std::literals](#)

Macros

- `#define __cpp_lib_string_udls`
- `#define _GLIBCXX_STRING_CONSTEXPR`

Functions

- `template<typename _InputIterator , typename _CharT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_CharT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::basic_string (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> basic_string< _CharT, char_traits< _CharT >, _Allocator >`

- `template<typename _CharT, typename _Traits, typename _Allocator = allocator<_CharT>, typename = _RequireAllocator<_Allocator>>
std::basic_string (basic_string_view< _CharT, _Traits >, const _Allocator &= _Allocator()) -> basic_string< _CharT, _Traits, _Allocator >`
- `template<typename _CharT, typename _Traits, typename _Allocator = allocator<_CharT>, typename = _RequireAllocator<_Allocator>>
std::basic_string (basic_string_view< _CharT, _Traits >, typename basic_string< _CharT, _Traits, _Allocator >::size_type, typename basic_string< _CharT, _Traits, _Allocator >::size_type, const _Allocator &= _Allocator())
-> basic_string< _CharT, _Traits, _Allocator >`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT, _Traits, _Alloc > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Alloc > &__str, _CharT __delim)`
- `template<> basic_istream< char > & std::getline (basic_istream< char > &__in, basic_string< char > &__str, char __delim)`
- `template<> basic_istream< wchar_t > & std::getline (basic_istream< wchar_t > &__in, basic_string< wchar_t > &__str, wchar_t __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
bool std::operator!= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
bool std::operator!= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
bool std::operator!= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `basic_string< char > std::literals::operator""s (const char *__str, size_t __len)`
- `basic_string< char16_t > std::literals::operator""s (const char16_t *__str, size_t __len)`
- `basic_string< char32_t > std::literals::operator""s (const char32_t *__str, size_t __len)`
- `basic_string< wchar_t > std::literals::operator""s (const wchar_t *__str, size_t __len)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`

- Generated by Doxygen

- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, basic_string< _CharT, _Traits, _Alloc > & __str)`
- `template<> basic_istream< char > & std::operator>> (basic_istream< char > & __is, basic_string< char > & __str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr void std::swap (basic_string< _CharT, _Traits, _Alloc > & __lhs, basic_string< _CharT, _Traits, _Alloc > & __rhs) noexcept(/*conditional */)`
- `string std::to_string (int __val)`
- `string std::to_string (long __val)`
- `string std::to_string (long long __val)`
- `string std::to_string (unsigned __val)`
- `string std::to_string (unsigned long __val)`
- `string std::to_string (unsigned long long __val)`

7.39.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

7.40 basic_string.tcc File Reference

Namespaces

- namespace `std`

Macros

- `#define _BASIC_STRING_TCC`
- `#define _GLIBCXX_STRING_CONSTEXPR`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > & __is, basic_string< _CharT, _Traits, _Alloc > & __str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`constexpr basic_string< _CharT, _Traits, _Alloc > std::operator+ (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, basic_string< _CharT, _Traits, _Alloc > & __str)`

7.40.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

7.41 boost_concept_check.h File Reference

Namespaces

- namespace [__gnu_cxx](#)
- namespace [__gnu_debug](#)
- namespace [std](#)

Macros

- `#define _GLIBCXX_CLASS_REQUIRES(_type_var, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES2(_type_var1, _type_var2, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES3(_type_var1, _type_var2, _type_var3, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES4(_type_var1, _type_var2, _type_var3, _type_var4, _ns, _concept)`
- `#define _GLIBCXX_DEFINE_BINARY_OPERATOR_CONSTRAINT(_OP, _NAME)`
- `#define _GLIBCXX_DEFINE_BINARY_PREDICATE_OP_CONSTRAINT(_OP, _NAME)`
- `#define _IsUnused`

Functions

- `template<class _Tp >`
`void __gnu_cxx::__aux_require_boolean_expr (const _Tp &__t)`
- `void __gnu_cxx::__error_type_must_be_a_signed_integer_type ()`
- `void __gnu_cxx::__error_type_must_be_an_integer_type ()`
- `void __gnu_cxx::__error_type_must_be_an_unsigned_integer_type ()`
- `template<class _Concept >`
`constexpr void __gnu_cxx::__function_requires ()`

7.41.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.42 c++0x_warning.h File Reference

7.42.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

7.43 char_traits.h File Reference

Classes

- struct [__gnu_cxx::Char_types<_CharT>](#)
- struct [__gnu_cxx::char_traits<_CharT>](#)
- struct [std::char_traits<_CharT>](#)
- struct [std::char_traits<char>](#)
- struct [std::char_traits<wchar_t>](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Macros

- `#define __cpp_lib_constexpr_char_traits`
- `#define _GLIBCXX_ALWAYS_INLINE`

7.43.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

7.44 charconv.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Functions

- `template<typename _Tp >`
`void std::__detail::__to_chars_10_impl (char *__first, unsigned __len, _Tp __val) noexcept`
- `template<typename _Tp >`
`constexpr unsigned std::__detail::__to_chars_len (_Tp __value, int __base) noexcept`

7.44.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<charconv>`.

7.45 chrono.h File Reference

Classes

- struct [std::common_type< chrono::duration< _Rep, _Period > >](#)
- struct [std::common_type< chrono::duration< _Rep, _Period >, chrono::duration< _Rep, _Period > >](#)
- struct [std::common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >](#)
- struct [std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >](#)
- class [std::chrono::duration< _Rep, _Period >](#)
- struct [std::chrono::duration_values< _Rep >](#)
- struct [std::chrono::_V2::steady_clock](#)
- struct [std::chrono::_V2::system_clock](#)
- class [std::chrono::time_point< _Clock, _Dur >](#)
- struct [std::chrono::treat_as_floating_point< _Rep >](#)

Namespaces

- namespace [std](#)
- namespace [std::chrono](#)
- namespace [std::filesystem](#)
- namespace [std::literals](#)
- namespace [std::literals::chrono_literals](#)

Macros

- `#define __cpp_lib_chrono`
- `#define __cpp_lib_chrono_udls`

Typedefs

- using [std::chrono::days](#) = [duration](#)< [int64_t](#), [ratio](#)< 86400 > >
- using [std::chrono::file_clock](#) = ::std::filesystem::__file_clock
- template<typename _Duration >
using [std::chrono::file_time](#) = [time_point](#)< [file_clock](#), _Duration >
- using [std::chrono::high_resolution_clock](#) = [system_clock](#)
- using [std::chrono::hours](#) = [duration](#)< [int64_t](#), [ratio](#)< 3600 > >
- using [std::chrono::microseconds](#) = [duration](#)< [int64_t](#), [micro](#) >
- using [std::chrono::milliseconds](#) = [duration](#)< [int64_t](#), [milli](#) >
- using [std::chrono::minutes](#) = [duration](#)< [int64_t](#), [ratio](#)< 60 > >
- using [std::chrono::months](#) = [duration](#)< [int64_t](#), [ratio](#)< 2629746 > >
- using [std::chrono::nanoseconds](#) = [duration](#)< [int64_t](#), [nano](#) >
- using [std::chrono::seconds](#) = [duration](#)< [int64_t](#) >
- using [std::chrono::sys_days](#) = [sys_time](#)< [days](#) >
- using [std::chrono::sys_seconds](#) = [sys_time](#)< [seconds](#) >
- template<typename _Duration >
using [std::chrono::sys_time](#) = [time_point](#)< [system_clock](#), _Duration >
- using [std::chrono::weeks](#) = [duration](#)< [int64_t](#), [ratio](#)< 604800 > >
- using [std::chrono::years](#) = [duration](#)< [int64_t](#), [ratio](#)< 31556952 > >

Functions

- template<typename _Rep, typename _Period >
constexpr [enable_if_t](#)< [numeric_limits](#)< _Rep >::is_signed, [duration](#)< _Rep, _Period > > [std::chrono::abs](#)
([duration](#)< _Rep, _Period > __d)
- template<typename _ToDur, typename _Rep, typename _Period >
constexpr __enable_if_is_duration< _ToDur > [std::chrono::ceil](#) (const [duration](#)< _Rep, _Period > &__d)
- template<typename _ToDur, typename _Clock, typename _Dur >
constexpr [enable_if_t](#)< __is_duration< _ToDur >::value, [time_point](#)< _Clock, _ToDur > > [std::chrono::ceil](#)
(const [time_point](#)< _Clock, _Dur > &__tp)
- template<typename _ToDur, typename _Rep, typename _Period >
constexpr __enable_if_is_duration< _ToDur > [std::chrono::duration_cast](#) (const [duration](#)< _Rep, _Period > &__d)
- template<typename _ToDur, typename _Rep, typename _Period >
constexpr __enable_if_is_duration< _ToDur > [std::chrono::floor](#) (const [duration](#)< _Rep, _Period > &__d)
- template<typename _ToDur, typename _Clock, typename _Dur >
constexpr [enable_if_t](#)< __is_duration< _ToDur >::value, [time_point](#)< _Clock, _ToDur > > [std::chrono::floor](#)
(const [time_point](#)< _Clock, _Dur > &__tp)

- `template<char... _Digits>`
`constexpr chrono::hours std::literals::chrono_literals::operator""h ()`
- `constexpr chrono::duration< long double, ratio< 3600, 1 > > std::literals::chrono_literals::operator""h (long double __hours)`
- `template<char... _Digits>`
`constexpr chrono::minutes std::literals::chrono_literals::operator""min ()`
- `constexpr chrono::duration< long double, ratio< 60, 1 > > std::literals::chrono_literals::operator""min (long double __mins)`
- `template<char... _Digits>`
`constexpr chrono::milliseconds std::literals::chrono_literals::operator""ms ()`
- `constexpr chrono::duration< long double, milli > std::literals::chrono_literals::operator""ms (long double __msecs)`
- `template<char... _Digits>`
`constexpr chrono::nanoseconds std::literals::chrono_literals::operator""ns ()`
- `constexpr chrono::duration< long double, nano > std::literals::chrono_literals::operator""ns (long double __nsecs)`
- `template<char... _Digits>`
`constexpr chrono::seconds std::literals::chrono_literals::operator""s ()`
- `constexpr chrono::duration< long double > std::literals::chrono_literals::operator""s (long double __secs)`
- `template<char... _Digits>`
`constexpr chrono::microseconds std::literals::chrono_literals::operator""us ()`
- `constexpr chrono::duration< long double, micro > std::literals::chrono_literals::operator""us (long double __usecs)`
- `template<typename _ToDur, typename _Rep, typename _Period >`
`constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename _ToDur::rep > > >::value, _ToDur > std::chrono::round (const duration< _Rep, _Period > &__d)`
- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr enable_if_t< __and< __is_duration< _ToDur >, __not< treat_as_floating_point< typename _ToDur::rep > > >::value, time_point< _Clock, _ToDur > > std::chrono::round (const time_point< _Clock, _Dur > &__tp)`
- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr __enable_if_t< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > > std::chrono::time_point_cast (const time_point< _Clock, _Dur > &__t)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::chrono::operator- (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > std::chrono::operator% (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type std::chrono::operator% (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Rep2, typename _Period >`
`constexpr duration< __common_rep_t< _Rep2, _Rep1 >, _Period > std::chrono::operator* (const _Rep1 &__s, const duration< _Rep2, _Period > &__d)`
- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< __common_rep_t< _Rep1, __disable_if_is_duration< _Rep2 > >, _Period > std::chrono::operator/ (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< _Rep1, _Rep2 >::type std::chrono::operator/ (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`

- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator!= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator< (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator<= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator> (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator>= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`

- `template<typename _Rep1, typename _Period1, typename _Clock, typename _Dur2 >`
`constexpr time_point< _Clock, typename common_type< duration< _Rep1, _Period1 >, _Dur2 >::type >`
`std::chrono::operator+ (const duration< _Rep1, _Period1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >`
`constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type >`
`std::chrono::operator- (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr common_type< _Dur1, _Dur2 >::type std::chrono::operator- (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`

- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator!= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator< (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator<= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator> (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator>= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`

Variables

- `template<typename _Tp >`
`constexpr bool std::chrono::is_clock_v`
- `template<> constexpr bool std::chrono::is_clock_v < file_clock >`

- `template<> constexpr bool std::chrono::is_clock_v< steady_clock >`
- `template<> constexpr bool std::chrono::is_clock_v< system_clock >`
- `template<typename _Rep >`
`constexpr bool std::chrono::treat_as_floating_point_v`

7.45.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<chrono>`.

7.46 `codecvt.h` File Reference

Classes

- class [`std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >`](#)
- class [`std::codecvt< _InternT, _ExternT, _StateT >`](#)
- class [`std::codecvt< char, char, mbstate_t >`](#)
- class [`std::codecvt< char16_t, char, mbstate_t >`](#)
- class [`std::codecvt< char32_t, char, mbstate_t >`](#)
- class [`std::codecvt< wchar_t, char, mbstate_t >`](#)
- class [`std::codecvt_base`](#)
- class [`std::codecvt_byname< _InternT, _ExternT, _StateT >`](#)

Namespaces

- namespace [`std`](#)

7.46.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.47 `concept_check.h` File Reference

Macros

- `#define __glibcxx_class_requires(_a, _b)`
- `#define __glibcxx_class_requires2(_a, _b, _c)`
- `#define __glibcxx_class_requires3(_a, _b, _c, _d)`
- `#define __glibcxx_class_requires4(_a, _b, _c, _d, _e)`
- `#define __glibcxx_function_requires(...)`

7.47.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.48 `cow_string.h` File Reference

Classes

- class [`std::basic_string< _CharT, _Traits, _Alloc >`](#)

Namespaces

- namespace [`std`](#)

Macros

- `#define __cpp_lib_constexpr_string`

7.48.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

Defines the reference-counted COW string implementation.

7.49 `cpp_type_traits.h` File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __INT_N(TYPE)`

Functions

- `template<typename _Iterator >
constexpr _Iterator std::__miter_base (_Iterator __it)`

7.49.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/type_traits>`.

7.50 `cxxabi_forced.h` File Reference

Classes

- class [__cxxabiv1::__forced_unwind](#)

7.50.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cxxabi.h>`.

7.51 `cxxabi_init_exception.h` File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CDTOR_CALLABI`
- `#define _GLIBCXX_HAVE_CDTOR_CALLABI`

Functions

- void * **__cxxabiv1::__cxa_allocate_exception** (size_t) noexcept
- void **__cxxabiv1::__cxa_free_exception** (void *) noexcept
- **__cxa_refcounted_exception** * **__cxxabiv1::__cxa_init_primary_exception** (void *object, [std::type_info](#) *tinfo, void(*dest)(void *)) noexcept

7.51.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.52 deque.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- **#define _DEQUE_TCC**

Functions

- template<bool _IsMove, typename _ITp, typename _IRef, typename _IPtr, typename _OTp >
::Deque_iterator< _OTp, _OTp &, _OTp * > **std::__copy_move_a1** (::Deque_iterator< _ITp, _IRef, _IPtr > __first, ::Deque_iterator< _ITp, _IRef, _IPtr > __last, ::Deque_iterator< _OTp, _OTp &, _OTp * > __result)
- template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
_OI **std::__copy_move_a1** (::Deque_iterator< _Tp, _Ref, _Ptr > __first, ::Deque_iterator< _Tp, _Ref, _Ptr > __last, _OI __result)
- template<bool _IsMove, typename _II, typename _Tp >
__gnu_cxx::__enable_if< __is_random_access_iter< _II >::__value, ::Deque_iterator< _Tp, _Tp &, _Tp * > >::__type **std::__copy_move_a1** (_II __first, _II __last, ::Deque_iterator< _Tp, _Tp &, _Tp * > __result)
- template<bool _IsMove, typename _CharT >
__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ::Deque_iterator< _CharT, _CharT &, _CharT * > >::__type **std::__copy_move_a2** (istreambuf_iterator< _CharT, [char_traits](#)< _CharT > > __first, istreambuf_iterator< _CharT, [char_traits](#)< _CharT > > __last, ::Deque_iterator< _CharT, _CharT &, _CharT * > __result)
- template<bool _IsMove, typename _ITp, typename _IRef, typename _IPtr, typename _OTp >
::Deque_iterator< _OTp, _OTp &, _OTp * > **std::__copy_move_backward_a1** (::Deque_iterator< _ITp, _IRef, _IPtr > __first, ::Deque_iterator< _ITp, _IRef, _IPtr > __last, ::Deque_iterator< _OTp, _OTp &, _OTp * > __result)
- template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
_OI **std::__copy_move_backward_a1** (::Deque_iterator< _Tp, _Ref, _Ptr > __first, ::Deque_iterator< _Tp, _Ref, _Ptr > __last, _OI __result)
- template<bool _IsMove, typename _II, typename _Tp >
__gnu_cxx::__enable_if< __is_random_access_iter< _II >::__value, ::Deque_iterator< _Tp, _Tp &, _Tp * > >::__type **std::__copy_move_backward_a1** (_II __first, _II __last, ::Deque_iterator< _Tp, _Tp &, _Tp * > __result)
- template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
_OI **std::__copy_move_backward_dit** (::Deque_iterator< _Tp, _Ref, _Ptr > __first, ::Deque_iterator< _Tp, _Ref, _Ptr > __last, _OI __result)
- template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >
_OI **std::__copy_move_dit** (::Deque_iterator< _Tp, _Ref, _Ptr > __first, ::Deque_iterator< _Tp, _Ref, _Ptr > __last, _OI __result)

- `template<typename _CharT, typename _Size >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, __Deque_iterator< _CharT, _CharT &, _CharT * >`
`>::__type std::__copy_n_a (istreambuf_iterator< _CharT, char_traits< _CharT > > __it, _Size __size, __`
`Deque_iterator< _CharT, _CharT &, _CharT * > __result, bool __strict)`
- `template<typename _Tp, typename _Ref, typename _Ptr, typename _Il >`
`__gnu_cxx::__enable_if< __is_random_access_iter< _Il >::__value, bool >::__type std::__equal_aux1 (`
`Deque_iterator< _Tp, _Ref, _Ptr > __first1, __Deque_iterator< _Tp, _Ref, _Ptr > __last1, _Il __first2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::__equal_aux1 (`
`Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, __Deque_iterator< _Tp1, _Ref1,`
`_Ptr1 > __last1, __Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2)`
- `template<typename _Il, typename _Tp, typename _Ref, typename _Ptr >`
`__gnu_cxx::__enable_if< __is_random_access_iter< _Il >::__value, bool >::__type std::__equal_aux1 (`
`_Il __first1, _Il __last1, __Deque_iterator< _Tp, _Ref, _Ptr > __first2)`
- `template<typename _Tp, typename _Ref, typename _Ptr, typename _Il >`
`bool std::__equal_dit (const __Deque_iterator< _Tp, _Ref, _Ptr > &__first1, const __Deque_iterator< _Tp,`
`_Ref, _Ptr > &__last1, _Il __first2)`
- `template<typename _Tp, typename _VTp >`
`void std::__fill_a1 (const __Deque_iterator< _Tp, _Tp &, _Tp * > &__first, const __Deque_iterator< _Tp, _Tp`
`&, _Tp * > &__last, const _VTp &__value)`
- `template<typename _Tp1, typename _Ref, typename _Ptr, typename _Tp2 >`
`int std::__lex_cmp_dit (`
`Deque_iterator< _Tp1, _Ref, _Ptr > __first1, __Deque_iterator< _Tp1, _Ref, _Ptr >`
`__last1, const _Tp2 *__first2, const _Tp2 *__last2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::__lexicographical_compare_aux1 (`
`Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, __Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, __Deque_iterator< _Tp2,`
`_Ref2, _Ptr2 > __first2, __Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2 >`
`bool std::__lexicographical_compare_aux1 (`
`Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, __Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, _Tp2 *__first2, _Tp2 *__last2)`
- `template<typename _Tp1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::__lexicographical_compare_aux1 (`
`_Tp1 *__first1, _Tp1 *__last1, __Deque_iterator< _Tp2, _Ref2,`
`_Ptr2 > __first2, __Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`

7.52.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<deque>`.

7.53 enable_special_members.h File Reference

Namespaces

- namespace [std](#)

7.53.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.54 erase_if.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_erase_if`

Functions

- `template<typename _Container, typename _UnsafeContainer, typename _Predicate >
_Container::size_type std::__detail::__erase_nodes_if (_Container &__cont, _UnsafeContainer &__ucont, _↵
Predicate __pred)`

7.54.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.55 `exception.h` File Reference

Classes

- class [std::exception](#)

Namespaces

- namespace [std](#)

7.55.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.56 `exception_defines.h` File Reference

Macros

- `#define __catch(X)`
- `#define __throw_exception_again`
- `#define __try`

7.56.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

7.57 `exception_ptr.h` File Reference

Classes

- class [std::__exception_ptr::exception_ptr](#)

Namespaces

- namespace [std](#)

Functions

- [exception_ptr std::current_exception](#) () noexcept
- `template<typename _Ex >
exception_ptr std::make_exception_ptr (_Ex __ex) noexcept`
- `void std::rethrow_exception (exception_ptr)`

7.57.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

7.58 forward_list.h File Reference

Classes

- struct [std::Fwd_list_base<_Tp, _Alloc>](#)
- struct [std::Fwd_list_const_iterator<_Tp>](#)
- struct [std::Fwd_list_iterator<_Tp>](#)
- struct [std::Fwd_list_node<_Tp>](#)
- struct [std::Fwd_list_node_base](#)
- class [std::forward_list<_Tp, _Alloc>](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_list_remove_return_type`
- `#define _GLIBCXX_FWLIST_REMOVE_RETURN_TYPE_TAG`

Functions

- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::forward_list(_InputIterator, _InputIterator, _Allocator=_Allocator()) -> forward_list<_ValT, _Allocator>`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator!=(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator<(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator<=(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator==(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator>(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator>=(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`void std::swap(forward_list<_Tp, _Alloc> &__lx, forward_list<_Tp, _Alloc> &__ly) noexcept(noexcept(__lx.swap(__ly)))`

7.58.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<forward_list>`.

7.59 forward_list.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _FORWARD_LIST_TCC`
- `#define _GLIBCXX20_ONLY(__expr)`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`

7.59.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<forward_list>`.

7.60 fs_dir.h File Reference

Classes

- struct `std::filesystem::__directory_iterator_proxy`
- class `std::filesystem::directory_entry`
- class `std::filesystem::directory_iterator`
- class `std::filesystem::file_status`
- class `std::filesystem::recursive_directory_iterator`

Namespaces

- namespace `std`
- namespace `std::filesystem`

7.60.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<filesystem>`.

7.61 fs_dir.h File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Functions

- `directory_iterator std::experimental::filesystem::begin (directory_iterator __iter) noexcept`
- `recursive_directory_iterator std::experimental::filesystem::begin (recursive_directory_iterator __iter) noexcept`
- `directory_iterator std::experimental::filesystem::end (directory_iterator) noexcept`
- `recursive_directory_iterator std::experimental::filesystem::end (recursive_directory_iterator) noexcept`
- `bool std::experimental::filesystem::operator!= (const directory_iterator &__lhs, const directory_iterator &__rhs)`
- `bool std::experimental::filesystem::operator!= (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)`
- `bool std::experimental::filesystem::operator== (const directory_iterator &__lhs, const directory_iterator &__rhs)`
- `bool std::experimental::filesystem::operator== (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)`

7.61.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

7.62 fs_fwd.h File Reference

Classes

- struct [std::filesystem::space_info](#)

Namespaces

- namespace [std](#)
- namespace [std::filesystem](#)

Typedefs

- using [std::filesystem::file_time_type](#) = [__file_clock::time_point](#)

Enumerations

- enum class [std::filesystem::copy_options](#) : unsigned short { **none**, **skip_existing**, **overwrite_existing**, **update_existing**, **recursive**, **copy_symlinks**, **skip_symlinks**, **directories_only**, **create_symlinks**, **create_hard_links** }
- enum class [std::filesystem::directory_options](#) : unsigned char { **none**, **follow_directory_symlink**, **skip_permission_denied** }
- enum class [std::filesystem::file_type](#) : signed char { **none**, **not_found**, **regular**, **directory**, **symlink**, **block**, **character**, **fifo**, **socket**, **unknown** }
- enum class [std::filesystem::perm_options](#) : unsigned { **replace**, **add**, **remove**, **nofollow** }
- enum class [std::filesystem::perms](#) : unsigned { **none**, **owner_read**, **owner_write**, **owner_exec**, **owner_all**, **group_read**, **group_write**, **group_exec**, **group_all**, **others_read**, **others_write**, **others_exec**, **others_all**, **all**, **set_uid**, **set_gid**, **sticky_bit**, **mask**, **unknown** }

Functions

- void [std::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options)
- void [std::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options, [error_code](#) &)
- bool [std::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option)
- bool [std::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option, [error_code](#) &)
- [path](#) [std::filesystem::current_path](#) ()
- bool [std::filesystem::exists](#) ([file_status](#)) noexcept
- [uintmax_t](#) [std::filesystem::file_size](#) (const [path](#) &)
- [uintmax_t](#) [std::filesystem::file_size](#) (const [path](#) &, [error_code](#) &) noexcept
- [uintmax_t](#) [std::filesystem::hard_link_count](#) (const [path](#) &)
- [uintmax_t](#) [std::filesystem::hard_link_count](#) (const [path](#) &, [error_code](#) &) noexcept
- bool [std::filesystem::is_other](#) ([file_status](#)) noexcept
- bool [std::filesystem::is_regular_file](#) ([file_status](#)) noexcept

- `bool std::filesystem::is_symlink (file_status) noexcept`
 - `file_time_type std::filesystem::last_write_time (const path &)`
 - `file_time_type std::filesystem::last_write_time (const path &, error_code &) noexcept`
 - `copy_options & std::filesystem::operator&= (copy_options &__x, copy_options __y) noexcept`
 - `constexpr copy_options std::filesystem::operator^ (copy_options __x, copy_options __y) noexcept`
 - `copy_options & std::filesystem::operator^= (copy_options &__x, copy_options __y) noexcept`
 - `constexpr copy_options std::filesystem::operator| (copy_options __x, copy_options __y) noexcept`
 - `copy_options & std::filesystem::operator|= (copy_options &__x, copy_options __y) noexcept`
 - `constexpr copy_options std::filesystem::operator~ (copy_options __x) noexcept`
 - `void std::filesystem::permissions (const path &, perms, perm_options, error_code &) noexcept`
 - `path std::filesystem::proximate (const path &__p, const path &__base, error_code &__ec)`
 - `path std::filesystem::relative (const path &__p, const path &__base, error_code &__ec)`
 - `bool std::filesystem::remove (const path &, error_code &) noexcept`
 - `uintmax_t std::filesystem::remove_all (const path &)`
 - `uintmax_t std::filesystem::remove_all (const path &, error_code &)`
 - `file_status std::filesystem::status (const path &)`
 - `file_status std::filesystem::status (const path &, error_code &) noexcept`
 - `bool std::filesystem::status_known (file_status) noexcept`
 - `file_status std::filesystem::symlink_status (const path &)`
 - `file_status std::filesystem::symlink_status (const path &, error_code &) noexcept`
-
- `perms & std::filesystem::operator&= (perms &__x, perms __y) noexcept`
 - `constexpr perms std::filesystem::operator^ (perms __x, perms __y) noexcept`
 - `perms & std::filesystem::operator^= (perms &__x, perms __y) noexcept`
 - `constexpr perms std::filesystem::operator| (perms __x, perms __y) noexcept`
 - `perms & std::filesystem::operator|= (perms &__x, perms __y) noexcept`
 - `constexpr perms std::filesystem::operator~ (perms __x) noexcept`
-
- `perm_options & std::filesystem::operator&= (perm_options &__x, perm_options __y) noexcept`
 - `constexpr perm_options std::filesystem::operator^ (perm_options __x, perm_options __y) noexcept`
 - `perm_options & std::filesystem::operator^= (perm_options &__x, perm_options __y) noexcept`
 - `constexpr perm_options std::filesystem::operator| (perm_options __x, perm_options __y) noexcept`
 - `perm_options & std::filesystem::operator|= (perm_options &__x, perm_options __y) noexcept`
 - `constexpr perm_options std::filesystem::operator~ (perm_options __x) noexcept`
-
- `directory_options & std::filesystem::operator&= (directory_options &__x, directory_options __y) noexcept`
 - `constexpr directory_options std::filesystem::operator^ (directory_options __x, directory_options __y) noexcept`
 - `directory_options & std::filesystem::operator^= (directory_options &__x, directory_options __y) noexcept`
 - `constexpr directory_options std::filesystem::operator| (directory_options __x, directory_options __y) noexcept`
 - `directory_options & std::filesystem::operator|= (directory_options &__x, directory_options __y) noexcept`
 - `constexpr directory_options std::filesystem::operator~ (directory_options __x) noexcept`

7.62.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<filesystem>`.

7.63 fs_fwd.h File Reference

Classes

- struct [std::experimental::filesystem::v1::space_info](#)

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Typedefs

- using [std::experimental::filesystem::file_time_type](#) = [std::chrono::system_clock::time_point](#)

Enumerations

- enum class [std::experimental::filesystem::copy_options](#) : unsigned short { **none**, **skip_existing**, **overwrite_existing**, **update_existing**, **recursive**, **copy_symlinks**, **skip_symlinks**, **directories_only**, **create_symlinks**, **create_hard_links** }
- enum class [std::experimental::filesystem::directory_options](#) : unsigned char { **none**, **follow_directory_symlink**, **skip_permission_denied** }
- enum class [std::experimental::filesystem::file_type](#) : signed char { **none**, **not_found**, **regular**, **directory**, **symlink**, **block**, **character**, **fifo**, **socket**, **unknown** }
- enum class [std::experimental::filesystem::perms](#) : unsigned { **none**, **owner_read**, **owner_write**, **owner_exec**, **owner_all**, **group_read**, **group_write**, **group_exec**, **group_all**, **others_read**, **others_write**, **others_exec**, **others_all**, **all**, **set_uid**, **set_gid**, **sticky_bit**, **mask**, **unknown**, **add_perms**, **remove_perms**, **symlink_nofollow** }

Functions

- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, copy_options __options)
- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, copy_options __options, [error_code](#) &) noexcept
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, copy_options __option)
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, copy_options __option, [error_code](#) &)
- [path](#) [std::experimental::filesystem::current_path](#) ()
- bool [std::experimental::filesystem::is_regular_file](#) (file_status) noexcept
- bool [std::experimental::filesystem::is_symlink](#) (file_status) noexcept
- copy_options & [std::experimental::filesystem::operator&=](#) (copy_options &__x, copy_options __y) noexcept
- constexpr copy_options [std::experimental::filesystem::operator^](#) (copy_options __x, copy_options __y) noexcept
- copy_options & [std::experimental::filesystem::operator^=](#) (copy_options &__x, copy_options __y) noexcept
- constexpr copy_options [std::experimental::filesystem::operator|](#) (copy_options __x, copy_options __y) noexcept
- copy_options & [std::experimental::filesystem::operator|=](#) (copy_options &__x, copy_options __y) noexcept
- constexpr copy_options [std::experimental::filesystem::operator~](#) (copy_options __x) noexcept

- file_status **std::experimental::filesystem::status** (const [path](#) &)
 - file_status **std::experimental::filesystem::status** (const [path](#) &, [error_code](#) &) noexcept
 - bool **std::experimental::filesystem::status_known** (file_status) noexcept
 - file_status **std::experimental::filesystem::symlink_status** (const [path](#) &)
 - file_status **std::experimental::filesystem::symlink_status** (const [path](#) &, [error_code](#) &) noexcept
-
- perms & **std::experimental::filesystem::operator&=** (perms &__x, perms __y) noexcept
 - constexpr perms **std::experimental::filesystem::operator^** (perms __x, perms __y) noexcept
 - perms & **std::experimental::filesystem::operator^=** (perms &__x, perms __y) noexcept
 - constexpr perms **std::experimental::filesystem::operator|** (perms __x, perms __y) noexcept
 - perms & **std::experimental::filesystem::operator|=** (perms &__x, perms __y) noexcept
 - constexpr perms **std::experimental::filesystem::operator~** (perms __x) noexcept
-
- directory_options & **std::experimental::filesystem::operator&=** (directory_options &__x, directory_options __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::operator^** (directory_options __x, directory_options __y) noexcept
 - directory_options & **std::experimental::filesystem::operator^=** (directory_options &__x, directory_options __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::operator|** (directory_options __x, directory_options __y) noexcept
 - directory_options & **std::experimental::filesystem::operator|=** (directory_options &__x, directory_options __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::operator~** (directory_options __x) noexcept

7.63.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

7.64 fs_ops.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::filesystem](#)

Functions

- [path](#) **std::filesystem::absolute** (const [path](#) &__p)
- [path](#) **std::filesystem::absolute** (const [path](#) &__p, [error_code](#) &__ec)
- [path](#) **std::filesystem::canonical** (const [path](#) &__p)
- [path](#) **std::filesystem::canonical** (const [path](#) &__p, [error_code](#) &__ec)
- void **std::filesystem::copy** (const [path](#) &__from, const [path](#) &__to)
- void **std::filesystem::copy** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options)
- void **std::filesystem::copy** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __options, [error_code](#) &__ec)
- void **std::filesystem::copy** (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec)
- bool **std::filesystem::copy_file** (const [path](#) &__from, const [path](#) &__to)
- bool **std::filesystem::copy_file** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option)
- bool **std::filesystem::copy_file** (const [path](#) &__from, const [path](#) &__to, [copy_options](#) __option, [error_code](#) &__ec)
- bool **std::filesystem::copy_file** (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec)

- void **std::filesystem::copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_symlink)
- void **std::filesystem::copy_symlink** (const [path](#) &__existing_symlink, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- bool **std::filesystem::create_directories** (const [path](#) &__p)
- bool **std::filesystem::create_directories** (const [path](#) &__p, [error_code](#) &__ec)
- bool **std::filesystem::create_directory** (const [path](#) &__p)
- bool **std::filesystem::create_directory** (const [path](#) &__p, const [path](#) &attributes)
- bool **std::filesystem::create_directory** (const [path](#) &__p, const [path](#) &attributes, [error_code](#) &__ec) noexcept
- bool **std::filesystem::create_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- void **std::filesystem::create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **std::filesystem::create_directory_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- void **std::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link)
- void **std::filesystem::create_hard_link** (const [path](#) &__to, const [path](#) &__new_hard_link, [error_code](#) &__ec) noexcept
- void **std::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink)
- void **std::filesystem::create_symlink** (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- [path](#) **std::filesystem::current_path** ()
- void **std::filesystem::current_path** (const [path](#) &__p)
- void **std::filesystem::current_path** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) **std::filesystem::current_path** ([error_code](#) &__ec)
- bool **std::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2)
- bool **std::filesystem::equivalent** (const [path](#) &__p1, const [path](#) &__p2, [error_code](#) &__ec) noexcept
- bool **std::filesystem::exists** (const [path](#) &__p)
- bool **std::filesystem::exists** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::exists** ([file_status](#)) noexcept
- [uintmax_t](#) **std::filesystem::file_size** (const [path](#) &__p)
- [uintmax_t](#) **std::filesystem::file_size** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [uintmax_t](#) **std::filesystem::hard_link_count** (const [path](#) &__p)
- [uintmax_t](#) **std::filesystem::hard_link_count** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_block_file** (const [path](#) &__p)
- bool **std::filesystem::is_block_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_block_file** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_character_file** (const [path](#) &__p)
- bool **std::filesystem::is_character_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_character_file** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_directory** (const [path](#) &__p)
- bool **std::filesystem::is_directory** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_directory** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_empty** (const [path](#) &__p)
- bool **std::filesystem::is_empty** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_fifo** (const [path](#) &__p)
- bool **std::filesystem::is_fifo** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_fifo** ([file_status](#) __s) noexcept
- bool **std::filesystem::is_other** (const [path](#) &__p)
- bool **std::filesystem::is_other** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_other** ([file_status](#)) noexcept
- bool **std::filesystem::is_regular_file** (const [path](#) &__p)
- bool **std::filesystem::is_regular_file** (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool **std::filesystem::is_regular_file** ([file_status](#)) noexcept

- `bool std::filesystem::is_socket (const path &__p)`
- `bool std::filesystem::is_socket (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_socket (file_status __s) noexcept`
- `bool std::filesystem::is_symlink (const path &__p)`
- `bool std::filesystem::is_symlink (const path &__p, error_code &__ec) noexcept`
- `bool std::filesystem::is_symlink (file_status) noexcept`
- `file_time_type std::filesystem::last_write_time (const path &)`
- `file_time_type std::filesystem::last_write_time (const path &, error_code &) noexcept`
- `void std::filesystem::last_write_time (const path &__p, file_time_type __new_time)`
- `void std::filesystem::last_write_time (const path &__p, file_time_type __new_time, error_code &__ec) noexcept`
- `void std::filesystem::permissions (const path &, perms, perm_options, error_code &) noexcept`
- `void std::filesystem::permissions (const path &__p, perms __prms, error_code &__ec) noexcept`
- `void std::filesystem::permissions (const path &__p, perms __prms, perm_options __opts=perm_options::replace)`
- `path std::filesystem::proximate (const path &__p, const path &__base, error_code &__ec)`
- `path std::filesystem::proximate (const path &__p, const path &__base=current_path())`
- `path std::filesystem::proximate (const path &__p, error_code &__ec)`
- `path std::filesystem::read_symlink (const path &__p)`
- `path std::filesystem::read_symlink (const path &__p, error_code &__ec)`
- `path std::filesystem::relative (const path &__p, const path &__base, error_code &__ec)`
- `path std::filesystem::relative (const path &__p, const path &__base=current_path())`
- `path std::filesystem::relative (const path &__p, error_code &__ec)`
- `bool std::filesystem::remove (const path &, error_code &) noexcept`
- `bool std::filesystem::remove (const path &__p)`
- `uintmax_t std::filesystem::remove_all (const path &)`
- `uintmax_t std::filesystem::remove_all (const path &, error_code &)`
- `void std::filesystem::rename (const path &__from, const path &__to)`
- `void std::filesystem::rename (const path &__from, const path &__to, error_code &__ec) noexcept`
- `void std::filesystem::resize_file (const path &__p, uintmax_t __size)`
- `void std::filesystem::resize_file (const path &__p, uintmax_t __size, error_code &__ec) noexcept`
- `space_info std::filesystem::space (const path &__p)`
- `space_info std::filesystem::space (const path &__p, error_code &__ec) noexcept`
- `file_status std::filesystem::status (const path &)`
- `file_status std::filesystem::status (const path &, error_code &) noexcept`
- `bool std::filesystem::status_known (file_status) noexcept`
- `file_status std::filesystem::symlink_status (const path &)`
- `file_status std::filesystem::symlink_status (const path &, error_code &) noexcept`
- `path std::filesystem::temp_directory_path ()`
- `path std::filesystem::temp_directory_path (error_code &__ec)`
- `path std::filesystem::weakly_canonical (const path &__p)`
- `path std::filesystem::weakly_canonical (const path &__p, error_code &__ec)`

7.64.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<filesystem>`.

7.65 fs_ops.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Functions

- [path std::experimental::filesystem::absolute](#) (const [path](#) &__p, const [path](#) &__base=current_path())
- [path std::experimental::filesystem::canonical](#) (const [path](#) &__p, const [path](#) &__base, [error_code](#) &__ec)
- [path std::experimental::filesystem::canonical](#) (const [path](#) &__p, const [path](#) &__base=current_path())
- [path std::experimental::filesystem::canonical](#) (const [path](#) &__p, [error_code](#) &__ec)
- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to)
- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, copy_options __options)
- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, copy_options __options, [error_code](#) &__ec) noexcept
- void [std::experimental::filesystem::copy](#) (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec) noexcept
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to)
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, copy_options __option)
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, copy_options __option, [error_code](#) &__ec)
- bool [std::experimental::filesystem::copy_file](#) (const [path](#) &__from, const [path](#) &__to, [error_code](#) &__ec)
- void [std::experimental::filesystem::copy_symlink](#) (const [path](#) &__existing_symlink, const [path](#) &__new_↔ symlink)
- void [std::experimental::filesystem::copy_symlink](#) (const [path](#) &__existing_symlink, const [path](#) &__new_↔ symlink, [error_code](#) &__ec) noexcept
- bool [std::experimental::filesystem::create_directories](#) (const [path](#) &__p)
- bool [std::experimental::filesystem::create_directories](#) (const [path](#) &__p, [error_code](#) &__ec)
- bool [std::experimental::filesystem::create_directory](#) (const [path](#) &__p)
- bool [std::experimental::filesystem::create_directory](#) (const [path](#) &__p, const [path](#) &attributes)
- bool [std::experimental::filesystem::create_directory](#) (const [path](#) &__p, const [path](#) &attributes, [error_code](#) &↔ __ec) noexcept
- bool [std::experimental::filesystem::create_directory](#) (const [path](#) &__p, [error_code](#) &__ec) noexcept
- void [std::experimental::filesystem::create_directory_symlink](#) (const [path](#) &__to, const [path](#) &__new_symlink)
- void [std::experimental::filesystem::create_directory_symlink](#) (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- void [std::experimental::filesystem::create_hard_link](#) (const [path](#) &__to, const [path](#) &__new_hard_link)
- void [std::experimental::filesystem::create_hard_link](#) (const [path](#) &__to, const [path](#) &__new_hard_link, [error_code](#) &__ec) noexcept
- void [std::experimental::filesystem::create_symlink](#) (const [path](#) &__to, const [path](#) &__new_symlink)
- void [std::experimental::filesystem::create_symlink](#) (const [path](#) &__to, const [path](#) &__new_symlink, [error_code](#) &__ec) noexcept
- [path std::experimental::filesystem::current_path](#) ()
- void [std::experimental::filesystem::current_path](#) (const [path](#) &__p)
- void [std::experimental::filesystem::current_path](#) (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path std::experimental::filesystem::current_path](#) ([error_code](#) &__ec)
- bool [std::experimental::filesystem::equivalent](#) (const [path](#) &__p1, const [path](#) &__p2)
- bool [std::experimental::filesystem::equivalent](#) (const [path](#) &__p1, const [path](#) &__p2, [error_code](#) &__ec) noexcept
- bool [std::experimental::filesystem::exists](#) (const [path](#) &__p)
- bool [std::experimental::filesystem::exists](#) (const [path](#) &__p, [error_code](#) &__ec) noexcept
- bool [std::experimental::filesystem::exists](#) (file_status __s) noexcept

- `uintmax_t std::experimental::filesystem::file_size` (const `path` & __p)
- `uintmax_t std::experimental::filesystem::file_size` (const `path` & __p, `error_code` & __ec) noexcept
- `uintmax_t std::experimental::filesystem::hard_link_count` (const `path` & __p)
- `uintmax_t std::experimental::filesystem::hard_link_count` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_block_file` (const `path` & __p)
- `bool std::experimental::filesystem::is_block_file` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_block_file` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_character_file` (const `path` & __p)
- `bool std::experimental::filesystem::is_character_file` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_character_file` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_directory` (const `path` & __p)
- `bool std::experimental::filesystem::is_directory` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_directory` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_empty` (const `path` & __p)
- `bool std::experimental::filesystem::is_empty` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_fifo` (const `path` & __p)
- `bool std::experimental::filesystem::is_fifo` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_fifo` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_other` (const `path` & __p)
- `bool std::experimental::filesystem::is_other` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_other` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_regular_file` (const `path` & __p)
- `bool std::experimental::filesystem::is_regular_file` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_regular_file` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_socket` (const `path` & __p)
- `bool std::experimental::filesystem::is_socket` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_socket` (file_status __s) noexcept
- `bool std::experimental::filesystem::is_symlink` (const `path` & __p)
- `bool std::experimental::filesystem::is_symlink` (const `path` & __p, `error_code` & __ec) noexcept
- `bool std::experimental::filesystem::is_symlink` (file_status __s) noexcept
- `file_time_type std::experimental::filesystem::last_write_time` (const `path` & __p)
- `file_time_type std::experimental::filesystem::last_write_time` (const `path` & __p, `error_code` & __ec) noexcept
- `void std::experimental::filesystem::last_write_time` (const `path` & __p, file_time_type __new_time)
- `void std::experimental::filesystem::last_write_time` (const `path` & __p, file_time_type __new_time, `error_code` & __ec) noexcept
- `void std::experimental::filesystem::permissions` (const `path` & __p, perms __prms)
- `void std::experimental::filesystem::permissions` (const `path` & __p, perms __prms, `error_code` & __ec) noexcept
- `path std::experimental::filesystem::read_symlink` (const `path` & __p)
- `path std::experimental::filesystem::read_symlink` (const `path` & __p, `error_code` & __ec)
- `bool std::experimental::filesystem::remove` (const `path` & __p)
- `bool std::experimental::filesystem::remove` (const `path` & __p, `error_code` & __ec) noexcept
- `uintmax_t std::experimental::filesystem::remove_all` (const `path` & __p)
- `uintmax_t std::experimental::filesystem::remove_all` (const `path` & __p, `error_code` & __ec)
- `void std::experimental::filesystem::rename` (const `path` & __from, const `path` & __to)
- `void std::experimental::filesystem::rename` (const `path` & __from, const `path` & __to, `error_code` & __ec) noexcept
- `void std::experimental::filesystem::resize_file` (const `path` & __p, uintmax_t __size)
- `void std::experimental::filesystem::resize_file` (const `path` & __p, uintmax_t __size, `error_code` & __ec) noexcept
- `space_info std::experimental::filesystem::space` (const `path` & __p)

- [space_info](#) `std::experimental::filesystem::space` (const [path](#) &__p, [error_code](#) &__ec) noexcept
- `file_status` `std::experimental::filesystem::status` (const [path](#) &__p)
- `file_status` `std::experimental::filesystem::status` (const [path](#) &__p, [error_code](#) &__ec) noexcept
- `bool` `std::experimental::filesystem::status_known` (file_status __s) noexcept
- `file_status` `std::experimental::filesystem::symlink_status` (const [path](#) &__p)
- `file_status` `std::experimental::filesystem::symlink_status` (const [path](#) &__p, [error_code](#) &__ec) noexcept
- [path](#) `std::experimental::filesystem::system_complete` (const [path](#) &__p)
- [path](#) `std::experimental::filesystem::system_complete` (const [path](#) &__p, [error_code](#) &__ec)
- [path](#) `std::experimental::filesystem::temp_directory_path` ()
- [path](#) `std::experimental::filesystem::temp_directory_path` ([error_code](#) &__ec)

7.65.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

7.66 fs_path.h File Reference

Classes

- class [std::filesystem::filesystem_error](#)
- class [std::filesystem::path::iterator](#)
- class [std::filesystem::path](#)

Namespaces

- namespace [std](#)
- namespace [std::filesystem](#)

Functions

- `size_t` `std::filesystem::hash_value` (const [path](#) &__p) noexcept

7.66.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<filesystem>`.

7.67 fs_path.h File Reference

Classes

- class [std::experimental::filesystem::v1::filesystem_error](#)
- class [std::experimental::filesystem::v1::path::iterator](#)
- class [std::experimental::filesystem::v1::path](#)

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Functions

- `bool` `std::experimental::filesystem::operator<` (const [path](#) &__lhs, const [path](#) &__rhs) noexcept
- `bool` `std::experimental::filesystem::operator==` (const [path](#) &__lhs, const [path](#) &__rhs) noexcept

7.67.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

7.68 `fstream.tcc` File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _FSTREAM_TCC`

7.68.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<fstream>`.

7.69 `functexcept.h` File Reference

Namespaces

- namespace [std](#)

Functions

- void **std::__throw_bad_alloc** (void)
- void **std::__throw_bad_array_new_length** (void)
- void **std::__throw_bad_cast** (void)
- void **std::__throw_bad_exception** (void)
- void **std::__throw_bad_function_call** ()
- void **std::__throw_bad_typeid** (void)
- void **std::__throw_domain_error** (const char *)
- void **std::__throw_future_error** (int)
- void **std::__throw_invalid_argument** (const char *)
- void **std::__throw_ios_failure** (const char *)
- void **std::__throw_ios_failure** (const char *, int)
- void **std::__throw_length_error** (const char *)
- void **std::__throw_logic_error** (const char *)
- void **std::__throw_out_of_range** (const char *)
- void **std::__throw_out_of_range_fmt** (const char *,...)
- void **std::__throw_overflow_error** (const char *)
- void **std::__throw_range_error** (const char *)
- void **std::__throw_runtime_error** (const char *)
- void **std::__throw_system_error** (int)
- void **std::__throw_underflow_error** (const char *)

7.69.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

This header provides support for `-fno-exceptions`.

7.70 functional_hash.h File Reference

Classes

- struct [std::hash< _Tp >](#)
- struct [std::hash< _Tp * >](#)
- struct [std::hash< bool >](#)
- struct [std::hash< char >](#)
- struct [std::hash< char16_t >](#)
- struct [std::hash< char32_t >](#)
- struct [std::hash< double >](#)
- struct [std::hash< float >](#)
- struct [std::hash< int >](#)
- struct [std::hash< long >](#)
- struct [std::hash< long double >](#)
- struct [std::hash< long long >](#)
- struct [std::hash< short >](#)
- struct [std::hash< signed char >](#)
- struct [std::hash< unsigned char >](#)
- struct [std::hash< unsigned int >](#)
- struct [std::hash< unsigned long >](#)
- struct [std::hash< unsigned long long >](#)
- struct [std::hash< unsigned short >](#)
- struct [std::hash< wchar_t >](#)

Namespaces

- namespace [std](#)

Macros

- `#define _Cxx_hashtable_define_trivial_hash(_Tp)`

7.70.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.71 gsllice.h File Reference

Classes

- class [std::gsllice](#)

Namespaces

- namespace [std](#)

7.71.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.72 `gslice_array.h` File Reference

Classes

- class `std::gslice_array<_Tp>`

Namespaces

- namespace `std`

7.72.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.73 `hash_bytes.h` File Reference

Namespaces

- namespace `std`

Functions

- `size_t std::Fnv_hash_bytes` (const void *__ptr, size_t __len, size_t __seed)
- `size_t std::Hash_bytes` (const void *__ptr, size_t __len, size_t __seed)

7.73.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.74 `hashtable.h` File Reference

Namespaces

- namespace `__gnu_cxx`

Enumerations

- enum { `_S_num_primes` }

Functions

- unsigned long `__gnu_cxx::__stl_next_prime` (unsigned long __n)
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All>`
`bool __gnu_cxx::operator!=` (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All>`
`bool __gnu_cxx::operator==` (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)
- `template<class _Val, class _Key, class _HF, class _Extract, class _EqKey, class _All>`
`void __gnu_cxx::swap` (hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht1, hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht2)

7.74.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.75 `hashtable.h` File Reference

Namespaces

- namespace [std](#)

7.75.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>` or `<unordered_set>`.

7.76 `hashtable_policy.h` File Reference

Namespaces

- namespace [std](#)

7.76.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>` or `<unordered_set>`.

7.77 `indirect_array.h` File Reference

Classes

- class [std::indirect_array<_Tp>](#)

Namespaces

- namespace [std](#)

7.77.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.78 `invoke.h` File Reference

Namespaces

- namespace [std](#)

Functions

- `template<typename _Tp, typename _Up = typename __inv_unwrap<_Tp>::type>
constexpr _Up && std::__invfwd (typename remove_reference<_Tp>::type &__t) noexcept`
- `template<typename _Callable, typename... _Args>
constexpr __invoke_result<_Callable, _Args...>::type std::__invoke (_Callable &&__fn, _Args &&... __args)
noexcept(__is_nothrow_invocable<_Callable, _Args...>::value)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>
constexpr _Res std::__invoke_impl (__invoke_memfun_deref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>
constexpr _Res std::__invoke_impl (__invoke_memfun_ref, _MemFun &&__f, _Tp &&__t, _Args &&... __args)`
- `template<typename _Res, typename _MemPtr, typename _Tp>
constexpr _Res std::__invoke_impl (__invoke_memobj_deref, _MemPtr &&__f, _Tp &&__t)`

- `template<typename _Res, typename _MemPtr, typename _Tp >`
`constexpr _Res std::__invoke_impl (__invoke_memobj_ref, _MemPtr &&__f, _Tp &&__t)`
- `template<typename _Res, typename _Fn, typename... _Args>`
`constexpr _Res std::__invoke_impl (__invoke_other, _Fn &&__f, _Args &&... __args)`
- `template<typename _Res, typename _Callable, typename... _Args>`
`constexpr enable_if_t< is_invocable_r_v< _Res, _Callable, _Args... >, _Res > std::__invoke_r (_Callable &&__fn, _Args &&... __args) noexcept(is_nothrow_invocable_r_v< _Res, _Callable, _Args... >)`

7.78.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.79 ios_base.h File Reference

Classes

- class [std::ios_base::failure](#)
- class [std::ios_base](#)

Namespaces

- namespace [std](#)

Enumerations

- enum `_ios_fmtflags` {
`_S_boolalpha`, `_S_dec`, `_S_fixed`, `_S_hex`,
`_S_internal`, `_S_left`, `_S_oct`, `_S_right`,
`_S_scientific`, `_S_showbase`, `_S_showpoint`, `_S_showpos`,
`_S_skipws`, `_S_unitbuf`, `_S_uppercase`, `_S_adjustfield`,
`_S_basefield`, `_S_floatfield`, `_S_ios_fmtflags_end`, `_S_ios_fmtflags_max`,
`_S_ios_fmtflags_min` }
- enum `_ios_iostate` {
`_S_goodbit`, `_S_badbit`, `_S_eofbit`, `_S_failbit`,
`_S_ios_iostate_end`, `_S_ios_iostate_max`, `_S_ios_iostate_min` }
- enum `_ios_Openmode` {
`_S_app`, `_S_ate`, `_S_bin`, `_S_in`,
`_S_out`, `_S_trunc`, `_S_noreplace`, `_S_ios_openmode_end`,
`_S_ios_openmode_max`, `_S_ios_openmode_min` }
- enum `_ios_Seekdir` { `_S_beg`, `_S_cur`, `_S_end`, `_S_ios_seekdir_end` }
- enum class [std::io_errc](#) { `stream` }

Functions

- `ios_base & std::boolalpha (ios_base &__base)`
- `ios_base & std::dec (ios_base &__base)`
- `ios_base & std::defaultfloat (ios_base &__base)`
- `ios_base & std::fixed (ios_base &__base)`
- `ios_base & std::hex (ios_base &__base)`
- `ios_base & std::hexfloat (ios_base &__base)`
- `ios_base & std::internal (ios_base &__base)`
- `const error_category & std::iostream_category ()` noexcept
- `ios_base & std::left (ios_base &__base)`

- [error_code](#) `std::make_error_code (io_errc __e)` noexcept
- [error_condition](#) `std::make_error_condition (io_errc __e)` noexcept
- [ios_base](#) & `std::noboolalpha (ios_base &__base)`
- [ios_base](#) & `std::noshowbase (ios_base &__base)`
- [ios_base](#) & `std::noshowpoint (ios_base &__base)`
- [ios_base](#) & `std::noshowpos (ios_base &__base)`
- [ios_base](#) & `std::noskipws (ios_base &__base)`
- [ios_base](#) & `std::nounitbuf (ios_base &__base)`
- [ios_base](#) & `std::nouppercase (ios_base &__base)`
- [ios_base](#) & `std::oct (ios_base &__base)`
- `constexpr _ios_Fmtflags std::operator& (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_losestate std::operator& (_ios_losestate __a, _ios_losestate __b)`
- `constexpr _ios_Openmode std::operator& (_ios_Openmode __a, _ios_Openmode __b)`
- `const _ios_Fmtflags & std::operator&= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_losestate & std::operator&= (_ios_losestate &__a, _ios_losestate __b)`
- `const _ios_Openmode & std::operator&= (_ios_Openmode &__a, _ios_Openmode __b)`
- `constexpr _ios_Fmtflags std::operator^ (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_losestate std::operator^ (_ios_losestate __a, _ios_losestate __b)`
- `constexpr _ios_Openmode std::operator^ (_ios_Openmode __a, _ios_Openmode __b)`
- `const _ios_Fmtflags & std::operator^= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_losestate & std::operator^= (_ios_losestate &__a, _ios_losestate __b)`
- `const _ios_Openmode & std::operator^= (_ios_Openmode &__a, _ios_Openmode __b)`
- `constexpr _ios_Fmtflags std::operator| (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_losestate std::operator| (_ios_losestate __a, _ios_losestate __b)`
- `constexpr _ios_Openmode std::operator| (_ios_Openmode __a, _ios_Openmode __b)`
- `const _ios_Fmtflags & std::operator|= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_losestate & std::operator|= (_ios_losestate &__a, _ios_losestate __b)`
- `const _ios_Openmode & std::operator|= (_ios_Openmode &__a, _ios_Openmode __b)`
- `constexpr _ios_Fmtflags std::operator~ (_ios_Fmtflags __a)`
- `constexpr _ios_losestate std::operator~ (_ios_losestate __a)`
- `constexpr _ios_Openmode std::operator~ (_ios_Openmode __a)`
- [ios_base](#) & `std::right (ios_base &__base)`
- [ios_base](#) & `std::scientific (ios_base &__base)`
- [ios_base](#) & `std::showbase (ios_base &__base)`
- [ios_base](#) & `std::showpoint (ios_base &__base)`
- [ios_base](#) & `std::showpos (ios_base &__base)`
- [ios_base](#) & `std::skipws (ios_base &__base)`
- [ios_base](#) & `std::unitbuf (ios_base &__base)`
- [ios_base](#) & `std::uppercase (ios_base &__base)`

7.79.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

7.80 istream.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _ISTREAM_TCC`

Functions

- `template<typename _CharT, typename _Traits >`
`void std::__istream_extract (basic_istream< _CharT, _Traits > &, _CharT *, streamsize)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::ws (basic_istream< _CharT, _Traits > &__is)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`

7.80.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<istream>`.

7.81 `iterator_concepts.h` File Reference

Classes

- struct [std::default_sentinel_t](#)
- struct [std::projected](#)< _Iter, _Proj >

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Concepts

- concept [std::indirectly_readable](#)
- concept [std::indirectly_writable](#)
- concept [std::weakly_incrementable](#)
- concept [std::indirectly_movable](#)
- concept [std::indirectly_copyable](#)
- concept [std::indirectly_swappable](#)
- concept [std::indirectly_comparable](#)
- concept [std::permutable](#)
- concept [std::mergeable](#)
- concept [std::sortable](#)

Typedefs

- `template<typename _Iter >`
`using std::__detail::__iter_concept = typename __iter_concept_impl< _Iter >::type`
- `template<typename _Tp >`
`using std::__detail::__iter_diff_t = typename __iter_traits< _Tp, incrementable_traits< _Tp > >::difference↵
_type`
- `template<typename _Iter, typename _Tp = _Iter>`
`using std::__detail::__iter_traits = typename __iter_traits_impl< _Iter, _Tp >::type`

- `template<typename _Tp >`
`using std::__detail::__iter_value_t = typename __iter_traits< _Tp, indirectly_readable_traits< _Tp > >`
`::value_type`
- `template<typename _Tp >`
`using std::__detail::__range_iter_t = decltype(ranges::__cust_access::__begin(std::declval< _Tp >()))`
- `template<typename _Tp >`
`using std::__detail::__with_ref = _Tp &`
- `template<typename _Fn, typename... _Is>`
`using std::indirect_result_t = invoke_result_t< _Fn, iter_reference_t< _Is >... >`
- `template<indirectly_readable _Tp>`
`using std::iter_common_reference_t = common_reference_t< iter_reference_t< _Tp >, iter_value_t< _Tp >`
`& >`
- `template<typename _Tp >`
`using std::iter_difference_t = __detail::__iter_diff_t< remove_cvref_t< _Tp > >`
- `template<__detail::__dereferenceable _Tp>`
`using std::iter_reference_t = decltype(*std::declval< _Tp >())`
- `template<__detail::__dereferenceable _Tp>`
`using std::iter_rvalue_reference_t = ranges::__cust_imove::__IMove::__type< _Tp >`
- `template<typename _Tp >`
`using std::iter_value_t = __detail::__iter_value_t< remove_cvref_t< _Tp > >`

Functions

- `template<typename _Tp >`
`requires is_array_v<_Tp> || __member_begin<_Tp&> || __adl_begin<_Tp&>`
`auto std::ranges::__cust_access::__begin (_Tp &__t)`
- `template<typename _Xp, typename _Yp >`
`constexpr iter_value_t< _Xp > std::ranges::__cust_iswap::__iter_exchange_move (_Xp &&__x, _Yp &&__y)`
`noexcept(noexcept(iter_value_t< _Xp > (iter_move(__x))) &&noexcept(*__x=iter_move(__y)))`
- `void std::ranges::__cust_access::__begin (auto &)=delete`
- `void std::ranges::__cust_access::__begin (const auto &)=delete`
- `void std::ranges::__cust_imove::iter_move ()`
- `template<typename _It1, typename _It2 >`
`void std::ranges::__cust_iswap::iter_swap (_It1, _It2)=delete`

Variables

- `struct std::ranges::__cust_access::Decay_copy std::ranges::__cust_access::__decay_copy`
- `constexpr default_sentinel_t std::default_sentinel`
- `template<typename _Sent, typename _Iter >`
`constexpr bool std::disable_sized_sentinel_for`
- `constexpr __cust_imove::__IMove std::ranges::iter_move`
- `constexpr __cust_iswap::__IterSwap std::ranges::iter_swap`
- `constexpr unreachable_sentinel_t std::unreachable_sentinel`

7.81.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.82 list.tcc File Reference

Namespaces

- namespace **std**

Macros

- `#define _GLIBCXX20_ONLY(__expr)`
- `#define _LIST_TCC`

7.82.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<list>`.

7.83 `locale_classes.h` File Reference

Classes

- class `std::collate<_CharT>`
- class `std::collate_byname<_CharT>`
- class `std::locale::facet`
- class `std::locale::id`
- class `std::locale`

Namespaces

- namespace `std`

7.83.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.84 `locale_classes.tcc` File Reference

Namespaces

- namespace `std`

Macros

- `#define _LOCALE_CLASSES_TCC`

Functions

- `template<typename _Facet>`
`bool std::has_facet (const locale &__loc) throw ()`
- `template<typename _Facet>`
`const _Facet & std::use_facet (const locale &__loc)`

7.84.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.85 `locale_conv.h` File Reference

Classes

- class `std::wbuffer_convert<_Codecvt, _Elem, _Tr>`
- class `std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>`

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Functions

- `template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >`
`bool std::__do_str_codecvt (const _InChar *__first, const _InChar *__last, _OutStr &__outstr, const _Codecvt &__cvt, _State &__state, size_t &__count, _Fn __fn)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in_all (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out_all (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`

7.85.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.86 locale_facets.h File Reference

Classes

- class [std::__ctype_abstract_base](#)< _CharT >
- class [std::ctype](#)< _CharT >
- class [std::ctype](#)< char >
- class [std::ctype](#)< wchar_t >
- class [std::ctype_byname](#)< _CharT >
- class [std::ctype_byname](#)< char >
- class [std::num_get](#)< _CharT, _InIter >
- class [std::num_put](#)< _CharT, _OutIter >
- class [std::num_punct](#)< _CharT >
- class [std::num_punct_byname](#)< _CharT >

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_NUM_CXX11_FACETS`
- `#define _GLIBCXX_NUM_FACETS`
- `#define _GLIBCXX_NUM_LBDL_ALT128_FACETS`
- `#define _GLIBCXX_NUM_UNICODE_FACETS`

Functions

- `template<typename _CharT >`
`_CharT * std::__add_grouping (_CharT * __s, _CharT __sep, const char * __gbeg, size_t __gsize, const _CharT`
`* __first, const _CharT * __last)`
- `template<typename _Tp >`
`void std::__convert_to_v (const char *, _Tp &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void std::__convert_to_v (const char *, double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void std::__convert_to_v (const char *, float &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<> void std::__convert_to_v (const char *, long double &, ios_base::iostate &, const __c_locale &)`
`throw ()`
- `template<typename _CharT, typename _Outlter >`
`_Outlter std::__write (_Outlter __s, const _CharT * __ws, int __len)`
- `template<typename _CharT >`
`ostreambuf_iterator< _CharT > std::__write (ostreambuf_iterator< _CharT > __s, const _CharT * __ws, int`
`__len)`
- `template<typename _CharT >`
`bool std::isalnum (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isalpha (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isblank (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::iscntrl (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isdigit (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isgraph (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::islower (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isprint (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::ispunct (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isspace (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isupper (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`bool std::isxdigit (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`_CharT std::tolower (_CharT __c, const locale & __loc)`
- `template<typename _CharT >`
`_CharT std::toupper (_CharT __c, const locale & __loc)`

7.86.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.87 locale_facets.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _LOCALE_FACETS_TCC`

Functions

- `template<typename _CharT >
_CharT * std::__add_grouping (_CharT *__s, _CharT __sep, const char *__gbeg, size_t __gsize, const _CharT *__first, const _CharT *__last)`
- `template<typename _CharT, typename _ValueT >
int std::__int_to_char (_CharT *__bufend, _ValueT __v, const _CharT *__lit, ios_base::fmtflags __flags, bool __dec)`
- `bool std::__verify_grouping (const char *__grouping, size_t __grouping_size, const string &__grouping_tmp) throw ()`

7.87.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.88 locale_facets_nonio.h File Reference

Classes

- class [std::messages<_CharT>](#)
- struct [std::messages_base](#)
- class [std::messages_byname<_CharT>](#)
- class [std::money_base](#)
- class [std::money_get<_CharT, _InIter>](#)
- class [std::money_put<_CharT, _OutIter>](#)
- class [std::moneypunct<_CharT, _Intl>](#)
- class [std::moneypunct_byname<_CharT, _Intl>](#)
- class [std::time_base](#)
- class [std::time_get<_CharT, _InIter>](#)
- class [std::time_get_byname<_CharT, _InIter>](#)
- class [std::time_put<_CharT, _OutIter>](#)
- class [std::time_put_byname<_CharT, _OutIter>](#)

Namespaces

- namespace [std](#)

7.88.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.89 locale_facets_nonio.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _LOCALE_FACETS_NONIO_TCC`

7.89.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.90 localefwd.h File Reference

Namespaces

- namespace [std](#)

Functions

- `template<typename _Facet >`
`bool std::has_facet (const locale &__loc) throw ()`
- `template<typename _CharT >`
`bool std::isalnum (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isalpha (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isblank (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::iscntrl (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isgraph (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::islower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isprint (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::ispunct (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isspace (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isupper (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isxdigit (_CharT __c, const locale &__loc)`

- `template<typename _CharT >`
`_CharT std::tolower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`_CharT std::toupper (_CharT __c, const locale &__loc)`
- `template<typename _Facet >`
`const _Facet & std::use_facet (const locale &__loc)`

7.90.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.91 mask_array.h File Reference

Classes

- class [std::mask_array<_Tp>](#)

Namespaces

- namespace [std](#)

7.91.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.92 max_size_type.h File Reference

7.92.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.93 memoryfwd.h File Reference

Namespaces

- namespace [std](#)

7.93.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.94 mofunc_impl.h File Reference

Classes

- class [std::move_only_function<_Res\(_ArgTypes...\) _GLIBCXX_MOF_CV noexcept\(_Noex\)>](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_MOF_CV_REF`
- `#define _GLIBCXX_MOF_INV_QUALS`
- `#define _GLIBCXX_MOF_REF`

7.94.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.95 move.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_addressof_constexpr`
- `#define _GLIBCXX_FORWARD(_Tp, __val)`
- `#define _GLIBCXX_FWDREF(_Tp)`
- `#define _GLIBCXX_MOVE(__val)`

Functions

- `template<typename _Tp >`
`constexpr _Tp * std::__addressof (_Tp &__r) noexcept`
- `template<typename _Tp, typename _Up = _Tp>`
`constexpr _Tp std::__exchange (_Tp &__obj, _Up &&__new_val)`
- `template<typename _Tp >`
`constexpr _Tp * std::addressof (_Tp &__r) noexcept`
- `template<typename _Tp >`
`const _Tp * std::addressof (const _Tp &&)=delete`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &&__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &__t) noexcept`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && std::move (_Tp &&__t) noexcept`
- `template<typename _Tp >`
`constexpr __conditional_t< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && > std::move_if_noexcept`
`(_Tp &__x) noexcept`
- `template<typename _Tp >`
`constexpr enable_if< __and_< __not_< __is_tuple_like< _Tp > >, is_move_constructible< _Tp >, is_move_assignable< _Tp > >::value >::type std::swap (_Tp &__a, _Tp &__b) noexcept(/*conditional */)`
`is_nothrow_move_assignable< _Tp > >`
- `template<typename _Tp, size_t _Nm>`
`constexpr enable_if< __is_swappable< _Tp >::value >::type std::swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])`
`noexcept(/*conditional */)`

7.95.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

7.96 move_only_function.h File Reference

7.96.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.97 nested_exception.h File Reference

Classes

- class [std::nested_exception](#)

Namespaces

- namespace [std](#)

Functions

- `template<typename _Ex >
void std::rethrow_if_nested (const _Ex &__ex)`
- `template<typename _Tp >
void std::throw_with_nested (_Tp &&__t)`

7.97.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

7.98 new_allocator.h File Reference

Classes

- class [std::__new_allocator<_Tp>](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_OPERATOR_DELETE`
- `#define _GLIBCXX_OPERATOR_NEW`
- `#define _GLIBCXX_SIZED_DEALLOC(p, n)`

7.98.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.99 new_allocator.h File Reference

Classes

- class [__gnu_cxx::new_allocator<_Tp>](#)

Namespaces

- namespace [__gnu_cxx](#)

7.99.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.100 node_handle.h File Reference

Classes

- class [std::_Node_handle<_Key, _Value, _NodeAlloc >](#)
- class [std::_Node_handle<_Value, _Value, _NodeAlloc >](#)
- class [std::_Node_handle_common<_Val, _NodeAlloc >](#)
- struct [std::_Node_insert_return<_Iterator, _NodeHandle >](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_node_extract`

7.100.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map,set,unordered_map,unordered_set>`.

7.101 ostream.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _OSTREAM_TCC`

Functions

- `template<typename _CharT, typename _Traits >`
[basic_ostream<_CharT, _Traits >](#) & [std::operator<<](#) ([basic_ostream<_CharT, _Traits >](#) &__out, const char *__s)

7.101.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ostream>`.

7.102 ostream_insert.h File Reference

Namespaces

- namespace [std](#)

Functions

- `template<typename _CharT, typename _Traits >`
`void std::__ostream_fill (basic_ostream< _CharT, _Traits > &__out, streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::__ostream_insert (basic_ostream< _CharT, _Traits > &__out, const _CharT * __s, streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`void std::__ostream_write (basic_ostream< _CharT, _Traits > &__out, const _CharT * __s, streamsize __n)`

7.102.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ostream>`.

7.103 `parse_numbers.h` File Reference

Namespaces

- namespace `std`

Typedefs

- `template<unsigned long long _Val>`
`using std::__parse_int::__ull_constant = integral_constant< unsigned long long, _Val >`
- `template<char... _Digs>`
`using std::__select_int::Select_int = typename _Select_int_base< __parse_int::Parse_int< _Digs... >::value, unsigned char, unsigned short, unsigned int, unsigned long, unsigned long long >::type`

7.103.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<chrono>`.

7.104 `postypes.h` File Reference

Classes

- class `std::fpos< _StateT >`

Namespaces

- namespace `std`

Typedefs

- `typedef long long std::streamoff`
- `typedef fpos< mbstate_t > std::streampos`
- `typedef ptrdiff_t std::streamsize`
- `typedef fpos< mbstate_t > std::u16streampos`
- `typedef fpos< mbstate_t > std::u32streampos`
- `typedef fpos< mbstate_t > std::wstreampos`

Functions

- `template<typename _StateT >`
`bool std::operator!= (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)`
- `template<typename _StateT >`
`bool std::operator== (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)`

7.104.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

7.105 predefined_ops.h File Reference

Namespaces

- namespace [__gnu_cxx](#)

Functions

- `template<typename _Compare >`
`constexpr _Iter_comp_iter< _Compare > __gnu_cxx::__ops::__iter_comp_iter (_Compare __comp)`
- `template<typename _Compare, typename _Iterator >`
`constexpr _Iter_comp_to_iter< _Compare, _Iterator > __gnu_cxx::__ops::__iter_comp_iter (_Iter_comp_↵
_iter< _Compare > __comp, _Iterator __it)`
- `template<typename _Iterator >`
`constexpr _Iter_equals_iter< _Iterator > __gnu_cxx::__ops::__iter_comp_iter (_Iter_equal_to_iter, _Iterator
__it)`
- `template<typename _Compare >`
`constexpr _Iter_comp_val< _Compare > __gnu_cxx::__ops::__iter_comp_val (_Compare __comp)`
- `template<typename _Compare, typename _Value >`
`_Iter_comp_to_val< _Compare, _Value > constexpr __gnu_cxx::__ops::__iter_comp_val (_Compare __↵
comp, _Value &__val)`
- `template<typename _Compare >`
`constexpr _Iter_comp_val< _Compare > __gnu_cxx::__ops::__iter_comp_val (_Iter_comp_iter< _Compare
> __comp)`
- `constexpr _Iter_equal_to_val __gnu_cxx::__ops::__iter_comp_val (_Iter_equal_to_iter)`
- `constexpr _Iter_less_val __gnu_cxx::__ops::__iter_comp_val (_Iter_less_iter)`
- `constexpr _Iter_equal_to_iter __gnu_cxx::__ops::__iter_equal_to_iter ()`
- `constexpr _Iter_equal_to_val __gnu_cxx::__ops::__iter_equal_to_val ()`
- `template<typename _Value >`
`constexpr _Iter_equals_val< _Value > __gnu_cxx::__ops::__iter_equals_val (_Value &__val)`
- `constexpr _Iter_less_iter __gnu_cxx::__ops::__iter_less_iter ()`
- `constexpr _Iter_less_val __gnu_cxx::__ops::__iter_less_val ()`
- `template<typename _Predicate >`
`constexpr _Iter_negate< _Predicate > __gnu_cxx::__ops::__negate (_Iter_pred< _Predicate > __pred)`
- `template<typename _Predicate >`
`constexpr _Iter_pred< _Predicate > __gnu_cxx::__ops::__pred_iter (_Predicate __pred)`
- `template<typename _Compare >`
`constexpr _Val_comp_iter< _Compare > __gnu_cxx::__ops::__val_comp_iter (_Compare __comp)`
- `template<typename _Compare >`
`constexpr _Val_comp_iter< _Compare > __gnu_cxx::__ops::__val_comp_iter (_Iter_comp_iter< _Compare
> __comp)`
- `constexpr _Val_less_iter __gnu_cxx::__ops::__val_comp_iter (_Iter_less_iter)`
- `constexpr _Val_less_iter __gnu_cxx::__ops::__val_less_iter ()`

7.105.1 Detailed Description

This is an internal header file, included by other library headers. You should not attempt to use it directly. Instead, include `<algorithm>`.

7.106 ptr_traits.h File Reference

Classes

- struct [std::pointer_traits<_Ptr>](#)
- struct [std::pointer_traits<_Tp*>](#)

Namespaces

- namespace [__gnu_debug](#)
- namespace [std](#)

Macros

- `#define __cpp_lib_constexpr_memory`
- `#define __cpp_lib_to_address`

Typedefs

- `template<typename _Ptr, typename _Tp>`
`using std::__ptr_rebind = typename pointer_traits<_Ptr>::template rebind<_Tp>`
- `template<typename _Ptr>`
`using std::__ptr_traits_elem_t = typename __get_first_arg<_Ptr>::type`

Functions

- `template<typename _Tp>`
`constexpr _Tp * std::__to_address (_Tp * __ptr) noexcept`
- `template<typename _Ptr>`
`constexpr auto std::__to_address (const _Ptr & __ptr) noexcept -> decltype(std::pointer_traits<_Ptr>::to_address(__ptr))`
- `template<typename _Ptr, typename... _None>`
`constexpr auto std::__to_address (const _Ptr & __ptr, _None...) noexcept`
- `template<typename _Tp>`
`constexpr _Tp * std::to_address (_Tp * __ptr) noexcept`
- `template<typename _Ptr>`
`constexpr auto std::to_address (const _Ptr & __ptr) noexcept`

7.106.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.107 quoted_string.h File Reference

Classes

- struct [std::__detail::__Quoted_string<_String, _CharT>](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Functions

- [template<typename _CharT, typename _Traits, typename _String >
std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< \(std::basic_ostream< _CharT, _Traits > &↵
__os, const _Quoted_string< _String, _CharT > &__str\)](#)
- [template<typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< \(std::basic_ostream< _CharT, _Traits > &↵
__os, const _Quoted_string< const _CharT *, _CharT > &__str\)](#)
- [template<typename _CharT, typename _Traits, typename _Alloc >
std::basic_istream< _CharT, _Traits > & std::__detail::operator>> \(std::basic_istream< _CharT, _Traits > &↵
__is, const _Quoted_string< basic_string< _CharT, _Traits, _Alloc > &, _CharT > &__str\)](#)

7.107.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iomanip>`.

7.108 random.h File Reference

Classes

- class [std::bernoulli_distribution](#)
- class [std::binomial_distribution< _IntType >](#)
- class [std::cauchy_distribution< _RealType >](#)
- class [std::chi_squared_distribution< _RealType >](#)
- class [std::discard_block_engine< _RandomNumberEngine, __p, __r >](#)
- class [std::discrete_distribution< _IntType >](#)
- class [std::exponential_distribution< _RealType >](#)
- class [std::extreme_value_distribution< _RealType >](#)
- class [std::fisher_f_distribution< _RealType >](#)
- class [std::gamma_distribution< _RealType >](#)
- class [std::geometric_distribution< _IntType >](#)
- class [std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >](#)
- class [std::linear_congruential_engine< _UIntType, __a, __c, __m >](#)
- class [std::lognormal_distribution< _RealType >](#)
- class [std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >](#)
- class [std::negative_binomial_distribution< _IntType >](#)
- class [std::normal_distribution< _RealType >](#)
- struct [std::exponential_distribution< _RealType >::param_type](#)
- struct [std::weibull_distribution< _RealType >::param_type](#)
- struct [std::poisson_distribution< _IntType >::param_type](#)
- struct [std::extreme_value_distribution< _RealType >::param_type](#)
- struct [std::negative_binomial_distribution< _IntType >::param_type](#)
- struct [std::discrete_distribution< _IntType >::param_type](#)
- struct [std::piecewise_constant_distribution< _RealType >::param_type](#)
- struct [std::geometric_distribution< _IntType >::param_type](#)
- struct [std::piecewise_linear_distribution< _RealType >::param_type](#)
- struct [std::student_t_distribution< _RealType >::param_type](#)

- struct `std::bernoulli_distribution::param_type`
- struct `std::uniform_real_distribution<_RealType>::param_type`
- struct `std::normal_distribution<_RealType>::param_type`
- struct `std::lognormal_distribution<_RealType>::param_type`
- struct `std::gamma_distribution<_RealType>::param_type`
- struct `std::chi_squared_distribution<_RealType>::param_type`
- struct `std::cauchy_distribution<_RealType>::param_type`
- struct `std::fisher_f_distribution<_RealType>::param_type`
- struct `std::binomial_distribution<_IntType>::param_type`
- class `std::piecewise_constant_distribution<_RealType>`
- class `std::piecewise_linear_distribution<_RealType>`
- class `std::poisson_distribution<_IntType>`
- class `std::random_device`
- class `std::seed_seq`
- class `std::shuffle_order_engine<_RandomNumberEngine, __k>`
- class `std::student_t_distribution<_RealType>`
- class `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>`
- class `std::uniform_real_distribution<_RealType>`
- class `std::weibull_distribution<_RealType>`

Namespaces

- namespace `std`

Typedefs

- typedef `minstd_rand0` `std::default_random_engine`
- typedef `shuffle_order_engine<minstd_rand0, 256>` `std::knuth_b`
- typedef `linear_congruential_engine<uint_fast32_t, 48271UL, 0UL, 2147483647UL>` `std::minstd_rand`
- typedef `linear_congruential_engine<uint_fast32_t, 16807UL, 0UL, 2147483647UL>` `std::minstd_rand0`
- typedef `mersenne_twister_engine<uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL>` `std::mt19937`
- typedef `mersenne_twister_engine<uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xfff7eee000000000ULL, 43, 6364136223846793005ULL>` `std::mt19937_64`
- typedef `discard_block_engine<ranlux24_base, 223, 23>` `std::ranlux24`
- typedef `subtract_with_carry_engine<uint_fast32_t, 24, 10, 24>` `std::ranlux24_base`
- typedef `discard_block_engine<ranlux48_base, 389, 11>` `std::ranlux48`
- typedef `subtract_with_carry_engine<uint_fast64_t, 48, 5, 12>` `std::ranlux48_base`

Functions

- template<typename `_RealType`, size_t `__bits`, typename `_UniformRandomNumberGenerator`>
`_RealType` `std::generate_canonical` (`_UniformRandomNumberGenerator` &`__g`)
- bool `std::operator!=` (const `std::bernoulli_distribution` &`__d1`, const `std::bernoulli_distribution` &`__d2`)
- template<typename `_IntType`>
bool `std::operator!=` (const `std::binomial_distribution<_IntType>` &`__d1`, const `std::binomial_distribution<_IntType>` &`__d2`)
- template<typename `_RealType`>
bool `std::operator!=` (const `std::cauchy_distribution<_RealType>` &`__d1`, const `std::cauchy_distribution<_RealType>` &`__d2`)

- `template<typename _RealType >`
`bool std::operator!= (const std::chi_squared_distribution< _RealType > &__d1, const std::chi_squared_distribution< _RealType > &__d2)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r>`
`bool std::operator!= (const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__lhs, const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__rhs)`
- `template<typename _IntType >`
`bool std::operator!= (const std::discrete_distribution< _IntType > &__d1, const std::discrete_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::fisher_f_distribution< _RealType > &__d1, const std::fisher_f_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::gamma_distribution< _RealType > &__d1, const std::gamma_distribution< _RealType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _IntType > &__d2)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType >`
`bool std::operator!= (const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__lhs, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__rhs)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>`
`bool std::operator!= (const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__lhs, const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__rhs)`
- `template<typename _RealType >`
`bool std::operator!= (const std::lognormal_distribution< _RealType > &__d1, const std::lognormal_distribution< _RealType > &__d2)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>`
`bool std::operator!= (const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__lhs, const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__rhs)`
- `template<typename _IntType >`
`bool std::operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2)`

- `template<typename _RandomNumberEngine, size_t __k>`
`bool std::operator!= (const std::shuffle_order_engine< _RandomNumberEngine, __k > &__lhs, const`
`std::shuffle_order_engine< _RandomNumberEngine, __k > &__rhs)`
- `template<typename _RealType >`
`bool std::operator!= (const std::student_t_distribution< _RealType > &__d1, const std::student_t_distribution<`
`_RealType > &__d2)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r>`
`bool std::operator!= (const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__lhs, const`
`std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__rhs)`
- `template<typename _IntType >`
`bool std::operator!= (const std::uniform_int_distribution< _IntType > &__d1, const std::uniform_int_distribution<`
`_IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::uniform_real_distribution< _IntType > &__d1, const std::uniform_real_distribution<`
`_IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _`
`RealType > &__d2)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const`
`std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const`
`std::uniform_real_distribution< _RealType > &)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::bernoulli_distribution &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::geometric_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::weibull_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &,`
`std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &,`
`std::uniform_real_distribution< _RealType > &)`
- `template<typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`std::bernoulli_distribution &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::weibull_distribution< _RealType > &__x)`

7.108.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

7.109 random.tcc File Reference

Namespaces

- namespace `std`
- namespace `std::__detail`

Macros

- `#define _RANDOM_TCC`

Functions

- `template<typename _ValT, typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & std::__detail::__extract_params (basic_istream< _CharT, _Traits > &__is,
vector< _ValT > &__vals, size_t __n)`
- `template<typename _Tp >
constexpr bool std::__detail::__p1_representable_as_double (_Tp __x) noexcept`
- `template<typename _Tp >
constexpr bool std::__detail::__representable_as_double (_Tp __x) noexcept`
- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >
_RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const
std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const
std::uniform_real_distribution< _RealType > &)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const binomial_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const chi_squared_distribution< _RealType > &__x)`

- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const discrete_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const gamma_distribution< _RealType > &__x)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const lognormal_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, ↵`
`_UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f`
`> &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const negative_binomial_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const normal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const poisson_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::bernoulli_distribution &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::extreme_value_distribution< _RealType > &__x)`

- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const student_t_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _RealType >`
`bool std::operator== (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &`
`std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &`
`std::uniform_real_distribution< _RealType > &)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, binomial_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, chi_squared_distribution< _RealType > &__x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, discrete_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, gamma_distribution< _RealType > &__x)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, lognormal_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s,`
`_UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, merseenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >`
`&__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`is, negative_binomial_distribution< _IntType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, normal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, poisson_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,
std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, student_t_distribution< _RealType > &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__↵
is, subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`

7.109.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

7.110 random.tcc File Reference

Namespaces

- namespace `__gnu_cxx`

Macros

- `#define _EXT_RANDOM_TCC`

Functions

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::beta_distribution< _RealType > &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::hypergeometric_distribution< _UIntType > &__x)`
- `template<size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__x)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t`
`__msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT`
`, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`&__os, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1,`
`__sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::triangular_distribution< _RealType > &__x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::uniform_inside_sphere_distribution< _Dimen, _RealType > &__x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const __gnu_cxx::von_mises_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const arcsine_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const hoyt_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const k_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const logistic_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const nakagami_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const pareto_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &←`
`__os, const rice_distribution< _RealType > &__x)`
- `template<size_t _Dimen, typename _RealType >`
`bool __gnu_cxx::operator== (const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d1, const`
`__gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d2)`

- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4>`
`bool gnu_cxx::operator== (const gnu_cxx::simd_fast_mersenne_twister_engine<_UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4> &__lhs, const gnu_cxx::simd_fast_mersenne_twister_engine<_UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4> &__rhs)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::beta_distribution<_RealType> &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::hypergeometric_distribution<_UIntType> &__x)`
- `template<size_t __Dimen, typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::normal_mv_distribution<__Dimen, _RealType> &__x)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::simd_fast_mersenne_twister_engine<_UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::triangular_distribution<_RealType> &__x)`
- `template<std::size_t __Dimen, typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::uniform_inside_sphere_distribution<__Dimen, _RealType> &__x)`
- `template<std::size_t __Dimen, typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::uniform_on_sphere_distribution<__Dimen, _RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, gnu_cxx::von_mises_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, arcsine_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, hoyt_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, k_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, logistic_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, nakagami_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, pareto_distribution<_RealType> &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & gnu_cxx::operator>> (std::basic_istream<_CharT, _Traits> &__is, rice_distribution<_RealType> &__x)`

7.110.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/random>`.

7.111 range_access.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_nonmember_container_access`
- `#define __cpp_lib_ssize`

Functions

- `template<typename _Container >`
`constexpr auto std::begin (_Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * std::begin (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::begin (const _Container &__cont) -> decltype(__cont.begin())`
- `template<class _Tp >`
`const _Tp * std::begin (const valarray< _Tp > &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::begin (valarray< _Tp > &__va) noexcept`
- `template<typename _Container >`
`constexpr auto std::cbegin (const _Container &__cont) noexcept(noexcept(std::begin(__cont))) -> decltype(std::begin(↵
__cont))`
- `template<typename _Container >`
`constexpr auto std::cend (const _Container &__cont) noexcept(noexcept(std::end(__cont))) -> decltype(std::end(↵
__cont))`
- `template<typename _Container >`
`constexpr auto std::crbegin (const _Container &__cont) -> decltype(std::rbegin(__cont))`
- `template<typename _Container >`
`constexpr auto std::crend (const _Container &__cont) -> decltype(std::rend(__cont))`
- `template<typename _Container >`
`constexpr auto std::data (_Container &__cont) noexcept(noexcept(__cont.data())) -> decltype(__cont.data())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * std::data (_Tp(&__array)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::data (const _Container &__cont) noexcept(noexcept(__cont.data())) -> decltype(__cont.↵
data())`
- `template<typename _Tp >`
`constexpr const _Tp * std::data (initializer_list< _Tp > __il) noexcept`
- `template<typename _Container >`
`constexpr auto std::empty (const _Container &__cont) noexcept(noexcept(__cont.empty())) -> decltype(__↵
cont.empty())`
- `template<typename _Tp, size_t _Nm>`
`constexpr bool std::empty (const _Tp(&)[_Nm]) noexcept`
- `template<typename _Tp >`
`constexpr bool std::empty (initializer_list< _Tp > __il) noexcept`

- `template<typename _Container >`
`constexpr auto std::end (_Container &__cont) -> decltype(__cont.end())`
- `template<typename _Tp, size_t _Nm>`
`constexpr _Tp * std::end (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::end (const _Container &__cont) -> decltype(__cont.end())`
- `template<class _Tp >`
`const _Tp * std::end (const valarray<_Tp> &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::end (valarray<_Tp> &__va) noexcept`
- `template<typename _Container >`
`constexpr auto std::rbegin (_Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp, size_t _Nm>`
`constexpr reverse_iterator<_Tp * > std::rbegin (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::rbegin (const _Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp >`
`constexpr reverse_iterator< const _Tp * > std::rbegin (initializer_list<_Tp> __il) noexcept`
- `template<typename _Container >`
`constexpr auto std::rend (_Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp, size_t _Nm>`
`constexpr reverse_iterator<_Tp * > std::rend (_Tp(&__arr)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::rend (const _Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp >`
`constexpr reverse_iterator< const _Tp * > std::rend (initializer_list<_Tp> __il) noexcept`
- `template<typename _Container >`
`constexpr auto std::size (const _Container &__cont) noexcept(noexcept(__cont.size())) -> decltype(__cont.size())`
- `template<typename _Tp, size_t _Nm>`
`constexpr size_t std::size (const _Tp(&)[_Nm]) noexcept`
- `template<typename _Container >`
`constexpr auto std::ssize (const _Container &__cont) noexcept(noexcept(__cont.size())) -> common_type_t<ptrdiff_t, make_signed_t< decltype(__cont.size())>>`
- `template<typename _Tp, ptrdiff_t _Num>`
`constexpr ptrdiff_t std::ssize (const _Tp(&)[_Num]) noexcept`

7.111.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.112 ranges_algo.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_shift`

Typedefs

- `template<typename _Iter1, typename _Iter2, typename _Out >`
`using std::ranges::binary_transform_result = in_in_out_result< _Iter1, _Iter2, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::copy_if_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Fp >`
`using std::ranges::for_each_n_result = in_fun_result< _Iter, _Fp >`
- `template<typename _Iter, typename _Fp >`
`using std::ranges::for_each_result = in_fun_result< _Iter, _Fp >`
- `template<typename _Iter1, typename _Iter2, typename _Out >`
`using std::ranges::merge_result = in_in_out_result< _Iter1, _Iter2, _Out >`
- `template<typename _Iter >`
`using std::ranges::minmax_element_result = min_max_result< _Iter >`
- `template<typename _Tp >`
`using std::ranges::minmax_result = min_max_result< _Tp >`
- `template<typename _Iter >`
`using std::ranges::next_permutation_result = in_found_result< _Iter >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::partial_sort_copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out1, typename _Out2 >`
`using std::ranges::partition_copy_result = in_out_out_result< _Iter, _Out1, _Out2 >`
- `template<typename _Iter >`
`using std::ranges::prev_permutation_result = in_found_result< _Iter >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::remove_copy_if_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::remove_copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::replace_copy_if_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::replace_copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::reverse_copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::rotate_copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::set_difference_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter1, typename _Iter2, typename _Out >`
`using std::ranges::set_intersection_result = in_in_out_result< _Iter1, _Iter2, _Out >`
- `template<typename _Iter1, typename _Iter2, typename _Out >`
`using std::ranges::set_symmetric_difference_result = in_in_out_result< _Iter1, _Iter2, _Out >`
- `template<typename _Iter1, typename _Iter2, typename _Out >`
`using std::ranges::set_union_result = in_in_out_result< _Iter1, _Iter2, _Out >`
- `template<typename _Iter1, typename _Iter2 >`
`using std::ranges::swap_ranges_result = in_in_result< _Iter1, _Iter2 >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::unary_transform_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::unique_copy_result = in_out_result< _Iter, _Out >`

Functions

- `template<typename _Comp, typename _Proj >`
`constexpr auto std::ranges::detail::__make_comp_proj (_Comp &__comp, _Proj &__proj)`
- `template<typename _Pred, typename _Proj >`
`constexpr auto std::ranges::detail::__make_pred_proj (_Pred &__pred, _Proj &__proj)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::shift_left (_ForwardIterator __first, _ForwardIterator __last, typename iterator_traits<_ForwardIterator >::difference_type __n)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::shift_right (_ForwardIterator __first, _ForwardIterator __last, typename iterator_traits<_ForwardIterator >::difference_type __n)`

Variables

- `constexpr __adjacent_find_fn std::ranges::adjacent_find`
- `constexpr __all_of_fn std::ranges::all_of`
- `constexpr __any_of_fn std::ranges::any_of`
- `constexpr __binary_search_fn std::ranges::binary_search`
- `constexpr __clamp_fn std::ranges::clamp`
- `constexpr __copy_if_fn std::ranges::copy_if`
- `constexpr __count_fn std::ranges::count`
- `constexpr __count_if_fn std::ranges::count_if`
- `constexpr __equal_range_fn std::ranges::equal_range`
- `constexpr __find_end_fn std::ranges::find_end`
- `constexpr __find_first_of_fn std::ranges::find_first_of`
- `constexpr __for_each_fn std::ranges::for_each`
- `constexpr __for_each_n_fn std::ranges::for_each_n`
- `constexpr __generate_fn std::ranges::generate`
- `constexpr __generate_n_fn std::ranges::generate_n`
- `constexpr __includes_fn std::ranges::includes`
- `constexpr __inplace_merge_fn std::ranges::inplace_merge`
- `constexpr __is_heap_fn std::ranges::is_heap`
- `constexpr __is_heap_until_fn std::ranges::is_heap_until`
- `constexpr __is_partitioned_fn std::ranges::is_partitioned`
- `constexpr __is_permutation_fn std::ranges::is_permutation`
- `constexpr __is_sorted_fn std::ranges::is_sorted`
- `constexpr __is_sorted_until_fn std::ranges::is_sorted_until`
- `constexpr __lexicographical_compare_fn std::ranges::lexicographical_compare`
- `constexpr __lower_bound_fn std::ranges::lower_bound`
- `constexpr __make_heap_fn std::ranges::make_heap`
- `constexpr __max_fn std::ranges::max`
- `constexpr __max_element_fn std::ranges::max_element`
- `constexpr __merge_fn std::ranges::merge`
- `constexpr __min_fn std::ranges::min`
- `constexpr __min_element_fn std::ranges::min_element`
- `constexpr __minmax_fn std::ranges::minmax`
- `constexpr __minmax_element_fn std::ranges::minmax_element`
- `constexpr __next_permutation_fn std::ranges::next_permutation`
- `constexpr __none_of_fn std::ranges::none_of`
- `constexpr __nth_element_fn std::ranges::nth_element`
- `constexpr __partial_sort_fn std::ranges::partial_sort`

- constexpr `__partial_sort_copy_fn` **`std::ranges::partial_sort_copy`**
- constexpr `__partition_fn` **`std::ranges::partition`**
- constexpr `__partition_copy_fn` **`std::ranges::partition_copy`**
- constexpr `__partition_point_fn` **`std::ranges::partition_point`**
- constexpr `__pop_heap_fn` **`std::ranges::pop_heap`**
- constexpr `__prev_permutation_fn` **`std::ranges::prev_permutation`**
- constexpr `__push_heap_fn` **`std::ranges::push_heap`**
- constexpr `__remove_fn` **`std::ranges::remove`**
- constexpr `__remove_copy_fn` **`std::ranges::remove_copy`**
- constexpr `__remove_copy_if_fn` **`std::ranges::remove_copy_if`**
- constexpr `__remove_if_fn` **`std::ranges::remove_if`**
- constexpr `__replace_fn` **`std::ranges::replace`**
- constexpr `__replace_copy_fn` **`std::ranges::replace_copy`**
- constexpr `__replace_copy_if_fn` **`std::ranges::replace_copy_if`**
- constexpr `__replace_if_fn` **`std::ranges::replace_if`**
- constexpr `__reverse_fn` **`std::ranges::reverse`**
- constexpr `__reverse_copy_fn` **`std::ranges::reverse_copy`**
- constexpr `__rotate_fn` **`std::ranges::rotate`**
- constexpr `__rotate_copy_fn` **`std::ranges::rotate_copy`**
- constexpr `__sample_fn` **`std::ranges::sample`**
- constexpr `__search_n_fn` **`std::ranges::search_n`**
- constexpr `__set_difference_fn` **`std::ranges::set_difference`**
- constexpr `__set_intersection_fn` **`std::ranges::set_intersection`**
- constexpr `__set_symmetric_difference_fn` **`std::ranges::set_symmetric_difference`**
- constexpr `__set_union_fn` **`std::ranges::set_union`**
- constexpr `__shuffle_fn` **`std::ranges::shuffle`**
- constexpr `__sort_fn` **`std::ranges::sort`**
- constexpr `__sort_heap_fn` **`std::ranges::sort_heap`**
- constexpr `__stable_partition_fn` **`std::ranges::stable_partition`**
- constexpr `__stable_sort_fn` **`std::ranges::stable_sort`**
- constexpr `__swap_ranges_fn` **`std::ranges::swap_ranges`**
- constexpr `__transform_fn` **`std::ranges::transform`**
- constexpr `__unique_fn` **`std::ranges::unique`**
- constexpr `__unique_copy_fn` **`std::ranges::unique_copy`**
- constexpr `__upper_bound_fn` **`std::ranges::upper_bound`**

7.112.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

7.113 `ranges_algobase.h` File Reference

Namespaces

- namespace `std`

Typedefs

- `template<typename _Iter1, typename _Iter2 >`
`using std::ranges::copy_backward_result = in_out_result< _Iter1, _Iter2 >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::copy_n_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::copy_result = in_out_result< _Iter, _Out >`
- `template<typename _Iter1, typename _Iter2 >`
`using std::ranges::move_backward_result = in_out_result< _Iter1, _Iter2 >`
- `template<typename _Iter, typename _Out >`
`using std::ranges::move_result = in_out_result< _Iter, _Out >`

Functions

- `template<bool _IsMove, input_iterator _Iter, sentinel_for< _Iter > _Sent, weakly_increamentable _Out>`
`requires (_IsMove ? indirectly_movable< _Iter, _Out> : indirectly_copyable< _Iter, _Out>)`
`constexpr __conditional_t< _IsMove, move_result< _Iter, _Out >, copy_result< _Iter, _Out > > std::ranges::`
`::__copy_or_move (_Iter __first, _Sent __last, _Out __result)`
- `template<bool _IsMove, bidirectional_iterator _Iter, sentinel_for< _Iter > _Sent, bidirectional_iterator _Out>`
`requires (_IsMove ? indirectly_movable< _Iter, _Out> : indirectly_copyable< _Iter, _Out>)`
`constexpr __conditional_t< _IsMove, move_backward_result< _Iter, _Out >, copy_backward_result< _Iter, _Out`
`> > std::ranges::__copy_or_move_backward (_Iter __first, _Sent __last, _Out __result)`

Variables

- `template<typename _Tp >`
`constexpr bool std::ranges::__detail::__is_move_iterator`
- `template<typename _Iterator >`
`constexpr bool std::ranges::__detail::__is_move_iterator< move_iterator< _Iterator > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::__detail::__is_normal_iterator`
- `template<typename _Tp >`
`constexpr bool std::ranges::__detail::__is_reverse_iterator`
- `template<typename _Iterator >`
`constexpr bool std::ranges::__detail::__is_reverse_iterator< reverse_iterator< _Iterator > >`
- `constexpr __copy_fn std::ranges::copy`
- `constexpr __copy_backward_fn std::ranges::copy_backward`
- `constexpr __copy_n_fn std::ranges::copy_n`
- `constexpr __equal_fn std::ranges::equal`
- `constexpr __fill_fn std::ranges::fill`
- `constexpr __fill_n_fn std::ranges::fill_n`
- `constexpr __move_fn std::ranges::move`
- `constexpr __move_backward_fn std::ranges::move_backward`

7.113.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

7.114 ranges_base.h File Reference

Classes

- struct [std::ranges::dangling](#)
- struct [std::ranges::view_base](#)

Namespaces

- namespace [std](#)

Concepts

- concept [std::ranges::range](#)
- concept [std::ranges::borrowed_range](#)
- concept [std::ranges::sized_range](#)
- concept [std::ranges::view](#)
- concept [std::ranges::output_range](#)
- concept [std::ranges::input_range](#)
- concept [std::ranges::forward_range](#)
- concept [std::ranges::bidirectional_range](#)
- concept [std::ranges::random_access_range](#)
- concept [std::ranges::contiguous_range](#)
- concept [std::ranges::common_range](#)
- concept [std::ranges::viewable_range](#)

Typedefs

- `template<typename _Tp >`
using **std::ranges::__detail::__make_unsigned_like_t** = decltype(__detail::__to_unsigned_like([std::declval](#)<_Tp >()))
- `template<range _Range>`
using **std::ranges::borrowed_iterator_t** = __conditional_t< borrowed_range< _Range >, iterator_t< _Range >, [dangling](#) >
- `template<typename _Tp >`
using **std::ranges::iterator_t** = std::__detail::__range_iter_t< _Tp >
- `template<range _Range>`
using **std::ranges::range_difference_t** = iter_difference_t< iterator_t< _Range > >
- `template<range _Range>`
using **std::ranges::range_reference_t** = iter_reference_t< iterator_t< _Range > >
- `template<range _Range>`
using **std::ranges::range_rvalue_reference_t** = iter_rvalue_reference_t< iterator_t< _Range > >
- `template<sized_range _Range>`
using **std::ranges::range_size_t** = decltype(ranges::size([std::declval](#)< _Range & >()))
- `template<range _Range>`
using **std::ranges::range_value_t** = iter_value_t< iterator_t< _Range > >
- `template<range _Range>`
using **std::ranges::sentinel_t** = decltype(ranges::end([std::declval](#)< _Range & >()))

Functions

- `template<typename _To , typename _Tp >`
constexpr decltype(auto) **std::ranges::__cust_access::__as_const** (_Tp &__t) noexcept
- `template<typename _Tp , typename _Up >`
requires (!same_as<_Tp, [view_interface](#)<_Up>>)
void **std::ranges::__detail::__is_derived_from_view_interface_fn** (const _Tp &, const [view_interface](#)<_Up > &)
- constexpr __max_size_type **std::ranges::__detail::__to_unsigned_like** (__max_diff_type __t) noexcept
- constexpr __max_size_type **std::ranges::__detail::__to_unsigned_like** (__max_size_type __t) noexcept
- `template<integral _Tp>`
constexpr auto **std::ranges::__detail::__to_unsigned_like** (_Tp __t) noexcept

- void **std::ranges::__cust_access::end** (auto &)=delete
- void **std::ranges::__cust_access::end** (const auto &)=delete
- void **std::ranges::__cust_access::rbegin** (auto &)=delete
- void **std::ranges::__cust_access::rbegin** (const auto &)=delete
- void **std::ranges::__cust_access::rend** (auto &)=delete
- void **std::ranges::__cust_access::rend** (const auto &)=delete
- void **std::ranges::__cust_access::size** (auto &)=delete
- void **std::ranges::__cust_access::size** (const auto &)=delete

Variables

- template<typename _Tp >
constexpr bool **std::ranges::__detail::__is_initializer_list**
- template<typename _Tp >
constexpr bool **std::ranges::__detail::__is_initializer_list**< **initializer_list**< _Tp > >
- constexpr __advance_fn **std::ranges::advance**
- constexpr __cust_access::_Begin **std::ranges::begin**
- constexpr __cust_access::_CBegin **std::ranges::cbegin**
- constexpr __cust_access::_CData **std::ranges::cdata**
- constexpr __cust_access::_CEnd **std::ranges::cend**
- constexpr __cust_access::_CRBegin **std::ranges::crbegin**
- constexpr __cust_access::_CREnd **std::ranges::crend**
- constexpr __cust_access::_Data **std::ranges::data**
- template<typename >
constexpr bool **std::ranges::disable_sized_range**
- constexpr __distance_fn **std::ranges::distance**
- constexpr __cust_access::_Empty **std::ranges::empty**
- template<typename _Tp >
constexpr bool **std::ranges::enable_borrowed_range**
- template<typename _Tp >
constexpr bool **std::ranges::enable_view**
- constexpr __cust_access::_End **std::ranges::end**
- constexpr __next_fn **std::ranges::next**
- constexpr __prev_fn **std::ranges::prev**
- constexpr __cust_access::_RBegin **std::ranges::rbegin**
- constexpr __cust_access::_REnd **std::ranges::rend**
- constexpr __cust_access::_Size **std::ranges::size**
- constexpr __cust_access::_SSize **std::ranges::ssize**

7.114.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ranges>`.

7.114.2 Variable Documentation

enable_view

```
template<typename _Tp >
constexpr bool std::ranges::enable_view [inline], [constexpr]
[range.view] The ranges::enable_view boolean.
```


7.115 ranges_cmp.h File Reference

Classes

- struct [std::ranges::equal_to](#)
- struct [std::ranges::greater](#)
- struct [std::ranges::greater_equal](#)
- struct [std::identity](#)
- struct [std::ranges::less](#)
- struct [std::ranges::less_equal](#)
- struct [std::ranges::not_equal_to](#)

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_ranges`

7.115.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.116 ranges_uninitialized.h File Reference

7.116.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.117 ranges_util.h File Reference

Classes

- class [std::ranges::subrange<_It, _Sent, _Kind>](#)
- class [std::ranges::view_interface<_Derived>](#)

Namespaces

- namespace [std](#)

Typedefs

- `template<range _Range>`
using `std::ranges::borrowed_subrange_t` = `__conditional_t< borrowed_range< _Range >, subrange< iterator_t< _Range > >, dangling >`
- `template<typename _Iter1, typename _Iter2>`
using `std::ranges::mismatch_result` = `in_in_result< _Iter1, _Iter2 >`

Enumerations

- enum class `subrange_kind` : `bool { unsized, sized }`

Functions

- `template<size_t _Num, class _It, class _Sent, subrange_kind _Kind>`
requires (`_Num < 2`)
`constexpr auto std::ranges::get (const subrange< _It, _Sent, _Kind > &__r)`
- `template<size_t _Num, class _It, class _Sent, subrange_kind _Kind>`
requires (`_Num < 2`)
`constexpr auto std::ranges::get (subrange< _It, _Sent, _Kind > &&__r)`
- `template<input_or_output_iterator _It, sentinel_for< _It > _Sent>`
`std::ranges::subrange (_It, _Sent) -> subrange< _It, _Sent >`
- `template<input_or_output_iterator _It, sentinel_for< _It > _Sent>`
`std::ranges::subrange (_It, _Sent, __detail::__make_unsigned_like_t< iter_difference_t< _It > >) ->`
`subrange< _It, _Sent, subrange_kind::sized >`
- `template<borrowed_range _Rng>`
`std::ranges::subrange (_Rng &&) -> subrange< iterator_t< _Rng >, sentinel_t< _Rng >, (sized_range< _Rng`
`>||sized_sentinel_for< sentinel_t< _Rng >, iterator_t< _Rng > >) ? subrange_kind::sized :subrange_kind::`
`unsized >`
- `template<borrowed_range _Rng>`
`std::ranges::subrange (_Rng &&, __detail::__make_unsigned_like_t< range_difference_t< _Rng > >) ->`
`subrange< iterator_t< _Rng >, sentinel_t< _Rng >, subrange_kind::sized >`

Variables

- `template<typename _It, typename _Sent, subrange_kind _Kind>`
`constexpr bool std::ranges::enable_borrowed_range< subrange< _It, _Sent, _Kind > >`
- `constexpr __find_fn std::ranges::find`
- `constexpr __find_if_fn std::ranges::find_if`
- `constexpr __find_if_not_fn std::ranges::find_if_not`
- `constexpr __mismatch_fn std::ranges::mismatch`
- `constexpr __search_fn std::ranges::search`

7.117.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ranges>`.

7.118 refwrap.h File Reference

Classes

- class `std::reference_wrapper< _Tp >`

Namespaces

- namespace `std`

Functions

- `template<typename _Tp >`
`std::reference_wrapper (_Tp &) -> reference_wrapper< _Tp >`

7.118.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.119 regex.h File Reference

Classes

- class [std::basic_regex< _Ch_type, _Rx_traits >](#)
- class [std::match_results< _Bi_iter, _Alloc >](#)
- class [std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >](#)
- class [std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >](#)
- class [std::regex_traits< _Ch_type >](#)
- class [std::sub_match< _Biliter >](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Typedefs

- typedef [match_results< const char * >](#) [std::cmatch](#)
- typedef [regex_iterator< const char * >](#) [std::cregex_iterator](#)
- typedef [regex_token_iterator< const char * >](#) [std::cregex_token_iterator](#)
- typedef [sub_match< const char * >](#) [std::csub_match](#)
- typedef [basic_regex< char >](#) [std::regex](#)
- typedef [match_results< string::const_iterator >](#) [std::smatch](#)
- typedef [regex_iterator< string::const_iterator >](#) [std::sregex_iterator](#)
- typedef [regex_token_iterator< string::const_iterator >](#) [std::sregex_token_iterator](#)
- typedef [sub_match< string::const_iterator >](#) [std::ssub_match](#)
- typedef [match_results< const wchar_t * >](#) [std::wcmatch](#)
- typedef [regex_iterator< const wchar_t * >](#) [std::wcregex_iterator](#)
- typedef [regex_token_iterator< const wchar_t * >](#) [std::wcregex_token_iterator](#)
- typedef [sub_match< const wchar_t * >](#) [std::wcs_sub_match](#)
- typedef [basic_regex< wchar_t >](#) [std::wregex](#)
- typedef [match_results< wstring::const_iterator >](#) [std::wsmatch](#)
- typedef [regex_iterator< wstring::const_iterator >](#) [std::wsregex_iterator](#)
- typedef [regex_token_iterator< wstring::const_iterator >](#) [std::wsregex_token_iterator](#)
- typedef [sub_match< wstring::const_iterator >](#) [std::wssub_match](#)

Enumerations

- enum class [_RegexExecutorPolicy](#) : int { [_S_auto](#) , [_S_alternate](#) }

Functions

- template<typename [_Biliter](#) , typename [_Alloc](#) , typename [_CharT](#) , typename [_TraitsT](#) >
bool [std::__detail::__regex_algo_impl](#) ([_Biliter](#) __s, [_Biliter](#) __e, [match_results< _Biliter, _Alloc >](#) &__m, const [basic_regex< _CharT, _TraitsT >](#) &__re, [regex_constants::match_flag_type](#) __flags, [_RegexExecutorPolicy](#) __policy, bool __match_mode)
- template<typename [_ForwardIterator](#) >
[std::basic_regex](#) ([_ForwardIterator](#), [_ForwardIterator](#), [regex_constants::syntax_option_type](#)={}) -> [basic_regex< typename iterator_traits< _ForwardIterator >::value_type >](#)
- template<typename [_Bi_iter](#) , class [_Alloc](#) >
bool [std::operator!=](#) (const [match_results< _Bi_iter, _Alloc >](#) &__m1, const [match_results< _Bi_iter, _Alloc >](#) &__m2)

- `template<typename _Bi_iter, typename _Alloc >`
`bool std::operator== (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`
- `template<typename _Bi_iter, typename _Alloc >`
`void std::swap (match_results< _Bi_iter, _Alloc > &__lhs, match_results< _Bi_iter, _Alloc > &__rhs) noexcept`

Matching, Searching, and Replacing

- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type * __fmt, size_t __len, regex_constants::match_flag_type __flags)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, class _Rx_traits >`
`bool std::regex_match (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Alloc, typename _Rx_traits >`
`bool std::regex_match (const _Ch_type * __s, match_results< const _Ch_type *, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Str_allocator > &__s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > & __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > & __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _Fst, _Fsa > & __fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`

- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const`
`basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m,`
`const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=`
`regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< type-`
`name basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _`
`Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< type-`
`name basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`

7.119.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.120 regex.tcc File Reference

Namespaces

- namespace `std`
- namespace `std::__detail`

7.120.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.121 regex_automaton.h File Reference

Classes

- class `std::__detail::_StateSeq< _TraitsT >`

Namespaces

- namespace `std`
- namespace `std::__detail`

Macros

- `#define _GLIBCXX_REGEX_STATE_LIMIT`

Typedefs

- `template<typename _CharT >`
`using std::__detail::Matcher = std::function< bool(_CharT)>`
- `typedef long std::__detail::StatelD`

Enumerations

- `enum std::__detail::Opcode : int {`
`_S_opcode_unknown , _S_opcode_alternative , _S_opcode_repeat , _S_opcode_backref ,`
`_S_opcode_line_begin_assertion , _S_opcode_line_end_assertion , _S_opcode_word_boundary , _S_←`
`opcode_subexpr_lookahead ,`
`_S_opcode_subexpr_begin , _S_opcode_subexpr_end , _S_opcode_dummy , _S_opcode_match ,`
`_S_opcode_accept }`

Variables

- `constexpr _StatelD std::__detail::S_invalid_state_id`

7.121.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.122 `regex_automaton.tcc` File Reference

Namespaces

- namespace `std`
- namespace `std::__detail`

7.122.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.123 `regex_compiler.h` File Reference

Classes

- `struct std::__detail::BracketMatcher< _TraitsT, __icase, __collate >`
- `class std::__detail::Compiler< _TraitsT >`

Namespaces

- namespace `std`
- namespace `std::__detail`

7.123.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.124 regex_compiler.tcc File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __INSERT_REGEX_MATCHER(__func, ...)`

7.124.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.125 regex_constants.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::regex_constants](#)

5.1 Regular Expression Syntax Options

- constexpr [syntax_option_type](#) [std::regex_constants::__multiline](#)
- constexpr [syntax_option_type](#) [std::regex_constants::__polynomial](#)
- constexpr [syntax_option_type](#) [std::regex_constants::awk](#)
- constexpr [syntax_option_type](#) [std::regex_constants::basic](#)
- constexpr [syntax_option_type](#) [std::regex_constants::collate](#)
- constexpr [syntax_option_type](#) [std::regex_constants::ECMAScript](#)
- constexpr [syntax_option_type](#) [std::regex_constants::egrep](#)
- constexpr [syntax_option_type](#) [std::regex_constants::extended](#)
- constexpr [syntax_option_type](#) [std::regex_constants::grep](#)
- constexpr [syntax_option_type](#) [std::regex_constants::icase](#)
- constexpr [syntax_option_type](#) [std::regex_constants::multiline](#)
- constexpr [syntax_option_type](#) [std::regex_constants::nosubs](#)
- constexpr [syntax_option_type](#) [std::regex_constants::operator&](#) ([syntax_option_type](#) __a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) & [std::regex_constants::operator&=](#) ([syntax_option_type](#) &__a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) [std::regex_constants::operator^](#) ([syntax_option_type](#) __a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) & [std::regex_constants::operator^=](#) ([syntax_option_type](#) &__a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) [std::regex_constants::operator|](#) ([syntax_option_type](#) __a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) & [std::regex_constants::operator|=](#) ([syntax_option_type](#) &__a, [syntax_option_type](#) __b)
- constexpr [syntax_option_type](#) [std::regex_constants::operator~](#) ([syntax_option_type](#) __a)
- constexpr [syntax_option_type](#) [std::regex_constants::optimize](#)
- enum [std::regex_constants::syntax_option_type](#) : unsigned int {
[_S_ica](#)se, [_S_nosubs](#), [_S_optimize](#), [_S_colla](#)te,
[_S_ECMAScript](#), [_S_ba](#)sic, [_S_exte](#)nded, [_S_awk](#),
[_S_grep](#), [_S_egrep](#), [_S_poly](#)nomial, [_S_multil](#)ine }

5.2 Matching Rules

Matching a regular expression against a sequence of characters [first, last) proceeds according to the rules of the grammar specified for the regular expression object, modified according to the effects listed below for any bitmask elements set.

- constexpr `match_flag_type` `std::regex_constants::format_default`
- constexpr `match_flag_type` `std::regex_constants::format_first_only`
- constexpr `match_flag_type` `std::regex_constants::format_no_copy`
- constexpr `match_flag_type` `std::regex_constants::format_sed`
- constexpr `match_flag_type` `std::regex_constants::match_any`
- constexpr `match_flag_type` `std::regex_constants::match_continuous`
- constexpr `match_flag_type` `std::regex_constants::match_default`
- enum `std::regex_constants::match_flag_type` : unsigned int {
`_S_default` , `_S_not_bol` , `_S_not_eol` , `_S_not_bow` ,
`_S_not_eow` , `_S_any` , `_S_not_null` , `_S_continuous` ,
`_S_prev_avail` , `_S_sed` , `_S_no_copy` , `_S_first_only` ,
`_S_match_flag_last` }
- constexpr `match_flag_type` `std::regex_constants::match_not_bol`
- constexpr `match_flag_type` `std::regex_constants::match_not_bow`
- constexpr `match_flag_type` `std::regex_constants::match_not_eol`
- constexpr `match_flag_type` `std::regex_constants::match_not_eow`
- constexpr `match_flag_type` `std::regex_constants::match_not_null`
- constexpr `match_flag_type` `std::regex_constants::match_prev_avail`
- constexpr `match_flag_type` `std::regex_constants::operator&` (`match_flag_type` __a, `match_flag_type` __b)
- constexpr `match_flag_type` & `std::regex_constants::operator&=` (`match_flag_type` &__a, `match_flag_type` __b)
- constexpr `match_flag_type` `std::regex_constants::operator^` (`match_flag_type` __a, `match_flag_type` __b)
- constexpr `match_flag_type` & `std::regex_constants::operator^=` (`match_flag_type` &__a, `match_flag_type` __b)
- constexpr `match_flag_type` `std::regex_constants::operator|` (`match_flag_type` __a, `match_flag_type` __b)
- constexpr `match_flag_type` & `std::regex_constants::operator|=` (`match_flag_type` &__a, `match_flag_type` __b)
- constexpr `match_flag_type` `std::regex_constants::operator~` (`match_flag_type` __a)

7.125.1 Detailed Description

Constant definitions for the std regex library.

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.126 `regex_error.h` File Reference

Classes

- class `std::regex_error`

Namespaces

- namespace `std`
- namespace `std::regex_constants`

5.3 Error Types

- constexpr [error_type](#) [std::regex_constants::error_backref](#) ([_S_error_backref](#))
- constexpr [error_type](#) [std::regex_constants::error_badbrace](#) ([_S_error_badbrace](#))
- constexpr [error_type](#) [std::regex_constants::error_badrepeat](#) ([_S_error_badrepeat](#))
- constexpr [error_type](#) [std::regex_constants::error_brace](#) ([_S_error_brace](#))
- constexpr [error_type](#) [std::regex_constants::error_brack](#) ([_S_error_brack](#))
- constexpr [error_type](#) [std::regex_constants::error_collate](#) ([_S_error_collate](#))
- constexpr [error_type](#) [std::regex_constants::error_complexity](#) ([_S_error_complexity](#))
- constexpr [error_type](#) [std::regex_constants::error_ctype](#) ([_S_error_ctype](#))
- constexpr [error_type](#) [std::regex_constants::error_escape](#) ([_S_error_escape](#))
- constexpr [error_type](#) [std::regex_constants::error_paren](#) ([_S_error_paren](#))
- constexpr [error_type](#) [std::regex_constants::error_range](#) ([_S_error_range](#))
- constexpr [error_type](#) [std::regex_constants::error_space](#) ([_S_error_space](#))
- constexpr [error_type](#) [std::regex_constants::error_stack](#) ([_S_error_stack](#))
- enum [std::regex_constants::error_type](#) {
 [_S_error_collate](#) , [_S_error_ctype](#) , [_S_error_escape](#) , [_S_error_backref](#) ,
 [_S_error_brack](#) , [_S_error_paren](#) , [_S_error_brace](#) , [_S_error_badbrace](#) ,
 [_S_error_range](#) , [_S_error_space](#) , [_S_error_badrepeat](#) , [_S_error_complexity](#) ,
 [_S_error_stack](#) , [_S_null](#) , [_S_grammar](#) }

7.126.1 Detailed Description

Error and exception objects for the std regex library.

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.127 `regex_executor.h` File Reference

Classes

- class [std::__detail::Executor](#)< [_Bilter](#) , [_Alloc](#) , [_TraitsT](#) , [__dfs_mode](#) >

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

7.127.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.128 `regex_executor.tcc` File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

7.128.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.129 `regex_scanner.h` File Reference

Classes

- class `std::__detail::_Scanner<_CharT>`

Namespaces

- namespace `std`
- namespace `std::__detail`

7.129.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.130 `regex_scanner.tcc` File Reference

Namespaces

- namespace `std`
- namespace `std::__detail`

7.130.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

7.131 `semaphore_base.h` File Reference

Namespaces

- namespace `std`

Typedefs

- using `std::__semaphore_impl` = `__atomic_semaphore`

7.131.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<semaphore>`.

7.132 `shared_ptr.h` File Reference

Classes

- class `std::enable_shared_from_this<_Tp>`
- struct `std::hash<shared_ptr<_Tp>>`
- struct `std::owner_less<shared_ptr<_Tp>>`
- struct `std::owner_less<void>`
- struct `std::owner_less<weak_ptr<_Tp>>`
- class `std::shared_ptr<_Tp>`
- class `std::weak_ptr<_Tp>`

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_enable_shared_from_this`
- `#define __cpp_lib_shared_ptr_weak_type`

Functions

- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * std::get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<typename _Tp, typename _Del >`
`std::shared_ptr (unique_ptr< _Tp, _Del >) -> shared_ptr< _Tp >`
- `template<typename _Tp >`
`std::shared_ptr (weak_ptr< _Tp >) -> shared_ptr< _Tp >`
- `template<typename _Tp >`
`std::weak_ptr (shared_ptr< _Tp >) -> weak_ptr< _Tp >`

7.132.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.133 `shared_ptr.h` File Reference

Classes

- struct [std::hash< experimental::shared_ptr< _Tp > >](#)
- struct [std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >](#)
- struct [std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >](#)

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Functions

- `template<typename _Tp >`
`bool std::experimental::atomic_compare_exchange_strong (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::experimental::atomic_compare_exchange_strong_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`bool std::experimental::atomic_compare_exchange_weak (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::experimental::atomic_compare_exchange_weak_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`void std::experimental::atomic_exchange (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`

- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::atomic_exchange_explicit (const shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r, memory_order __mo)`
- `template<typename _Tp >`
`bool std::experimental::atomic_is_lock_free (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::atomic_load (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::atomic_load_explicit (const shared_ptr< _Tp > *__p, memory_order __mo)`
- `template<typename _Tp >`
`void std::experimental::atomic_store (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::atomic_store_explicit (const shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r, memory_order __mo)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::const_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::dynamic_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Del, typename _Tp >`
`_Del * std::experimental::get_deleter (const shared_ptr< _Tp > &__p) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Ch, typename _Tr, typename _Tp >`
`std::basic_ostream< _Ch, _Tr > & std::experimental::operator<< (std::basic_ostream< _Ch, _Tr > &__os, const shared_ptr< _Tp > &__p)`
- `template<typename _Tp >`
`bool std::experimental::operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`

- `template<typename _Tp >`
`bool std::experimental::operator> (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator> (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator> (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::reinterpret_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::static_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp >`
`void std::experimental::swap (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp >`
`void std::experimental::swap (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept`

Variables

- `template<typename _Yp, typename _Tp >`
`constexpr bool std::experimental::__sp_compatible_v`
- `template<typename _Tp, typename _Yp >`
`constexpr bool std::experimental::__sp_is_constructible_v`

7.133.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/memory>`.

7.134 shared_ptr_atomic.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_atomic_shared_ptr`
- `#define _GLIBCXX_TSAN_MUTEX_DESTROY(X)`
- `#define _GLIBCXX_TSAN_MUTEX_LOCKED(X)`
- `#define _GLIBCXX_TSAN_MUTEX_POST_SIGNAL(X)`
- `#define _GLIBCXX_TSAN_MUTEX_POST_UNLOCK(X)`
- `#define _GLIBCXX_TSAN_MUTEX_PRE_SIGNAL(X)`
- `#define _GLIBCXX_TSAN_MUTEX_PRE_UNLOCK(X)`
- `#define _GLIBCXX_TSAN_MUTEX_TRY_LOCK(X)`
- `#define _GLIBCXX_TSAN_MUTEX_TRY_LOCK_FAILED(X)`

Variables

- template<typename _Up >
static constexpr bool **std::__is_shared_ptr**
- template<typename _Up >
static constexpr bool **std::__is_shared_ptr< shared_ptr< _Up > >**

7.134.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include <memory>.

7.135 shared_ptr_base.h File Reference

Classes

- struct [std::Sp_ebo_helper< _Nm, _Tp, false >](#)
- struct [std::Sp_ebo_helper< _Nm, _Tp, true >](#)
- class [std::bad_weak_ptr](#)
- struct [std::hash< __shared_ptr< _Tp, _Lp > >](#)

Namespaces

- namespace [std](#)

Macros

- #define **__cpp_lib_shared_ptr_arrays**
- #define **__cpp_lib_smart_ptr_for_overwrite**

Functions

- template<typename _Tp, _Lock_policy _Lp = __default_lock_policy, typename _Alloc, typename... _Args>
[__shared_ptr< _Tp, _Lp >](#) **std::allocate_shared** (const _Alloc &__a, _Args &&... __args)
- template<typename _Tp, _Lock_policy _Lp = __default_lock_policy, typename... _Args>
[__shared_ptr< _Tp, _Lp >](#) **std::make_shared** (_Args &&... __args)
- void **std::throw_bad_weak_ptr** ()
- template<typename _Tp, typename _Tp1, _Lock_policy _Lp>
[__shared_ptr< _Tp, _Lp >](#) **std::const_pointer_cast** (const [__shared_ptr< _Tp1, _Lp >](#) &__r) noexcept
- template<typename _Tp, typename _Tp1, _Lock_policy _Lp>
[__shared_ptr< _Tp, _Lp >](#) **std::dynamic_pointer_cast** (const [__shared_ptr< _Tp1, _Lp >](#) &__r) noexcept
- template<typename _Tp, _Lock_policy _Lp>
bool **std::operator!=** (const [__shared_ptr< _Tp, _Lp >](#) &__a, nullptr_t) noexcept
- template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>
bool **std::operator!=** (const [__shared_ptr< _Tp1, _Lp >](#) &__a, const [__shared_ptr< _Tp2, _Lp >](#) &__b) noexcept
- template<typename _Tp, _Lock_policy _Lp>
bool **std::operator!=** (nullptr_t, const [__shared_ptr< _Tp, _Lp >](#) &__a) noexcept
- template<typename _Tp, typename _Up, _Lock_policy _Lp>
bool **std::operator<** (const [__shared_ptr< _Tp, _Lp >](#) &__a, const [__shared_ptr< _Up, _Lp >](#) &__b) noexcept
- template<typename _Tp, _Lock_policy _Lp>
bool **std::operator<** (const [__shared_ptr< _Tp, _Lp >](#) &__a, nullptr_t) noexcept
- template<typename _Tp, _Lock_policy _Lp>
bool **std::operator<** (nullptr_t, const [__shared_ptr< _Tp, _Lp >](#) &__a) noexcept

- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator<= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator<= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator<= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator== (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator== (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator== (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator> (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator> (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator> (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator>= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator>= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator>= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::reinterpret_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::static_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::swap (__shared_ptr< _Tp, _Lp > &__a, __shared_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::swap (__weak_ptr< _Tp, _Lp > &__a, __weak_ptr< _Tp, _Lp > &__b) noexcept`

7.135.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.136 `slice_array.h` File Reference

Classes

- class `std::slice`
- class `std::slice_array< _Tp >`

Namespaces

- namespace `std`

7.136.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.137 specfun.h File Reference

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Macros

- `#define __cpp_lib_math_special_functions`
- `#define __STDCPP_MATH_SPEC_FUNCS__`

Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_ai (_Tp __x)`
- `float __gnu_cxx::airy_aif (float __x)`
- `long double __gnu_cxx::airy_ail (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type __gnu_cxx::airy_bi (_Tp __x)`
- `float __gnu_cxx::airy_bif (float __x)`
- `long double __gnu_cxx::airy_bil (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float std::assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double std::assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float std::assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double std::assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type std::beta (_Tpa __a, _Tpb __b)`
- `float std::betaf (float __a, float __b)`
- `long double std::betal (long double __a, long double __b)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_1 (_Tp __k)`
- `float std::comp_ellint_1f (float __k)`
- `long double std::comp_ellint_1l (long double __k)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_2 (_Tp __k)`
- `float std::comp_ellint_2f (float __k)`
- `long double std::comp_ellint_2l (long double __k)`
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::comp_ellint_3 (_Tp __k, _Tpn __nu)`
- `float std::comp_ellint_3f (float __k, float __nu)`
- `long double std::comp_ellint_3l (long double __k, long double __nu)`
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type __gnu_cxx::conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x)`

- float [__gnu_cxx::conf_hypergf](#) (float __a, float __c, float __x)
- long double [__gnu_cxx::conf_hypergl](#) (long double __a, long double __c, long double __x)
- template<typename _Tpnu, typename _Tp >
 [__gnu_cxx::__promote_2](#)< _Tpnu, _Tp >::__type [std::cyl_bessel_i](#) (_Tpnu __nu, _Tp __x)
- float [std::cyl_bessel_if](#) (float __nu, float __x)
- long double [std::cyl_bessel_il](#) (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
 [__gnu_cxx::__promote_2](#)< _Tpnu, _Tp >::__type [std::cyl_bessel_j](#) (_Tpnu __nu, _Tp __x)
- float [std::cyl_bessel_jf](#) (float __nu, float __x)
- long double [std::cyl_bessel_jl](#) (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
 [__gnu_cxx::__promote_2](#)< _Tpnu, _Tp >::__type [std::cyl_bessel_k](#) (_Tpnu __nu, _Tp __x)
- float [std::cyl_bessel_kf](#) (float __nu, float __x)
- long double [std::cyl_bessel_kl](#) (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
 [__gnu_cxx::__promote_2](#)< _Tpnu, _Tp >::__type [std::cyl_neumann](#) (_Tpnu __nu, _Tp __x)
- float [std::cyl_neumannf](#) (float __nu, float __x)
- long double [std::cyl_neumannl](#) (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
 [__gnu_cxx::__promote_2](#)< _Tp, _Tpp >::__type [std::ellint_1](#) (_Tp __k, _Tpp __phi)
- float [std::ellint_1f](#) (float __k, float __phi)
- long double [std::ellint_1l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
 [__gnu_cxx::__promote_2](#)< _Tp, _Tpp >::__type [std::ellint_2](#) (_Tp __k, _Tpp __phi)
- float [std::ellint_2f](#) (float __k, float __phi)
- long double [std::ellint_2l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
 [__gnu_cxx::__promote_3](#)< _Tp, _Tpn, _Tpp >::__type [std::ellint_3](#) (_Tp __k, _Tpn __nu, _Tpp __phi)
- float [std::ellint_3f](#) (float __k, float __nu, float __phi)
- long double [std::ellint_3l](#) (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
 [__gnu_cxx::__promote](#)< _Tp >::__type [std::expint](#) (_Tp __x)
- float [std::expintf](#) (float __x)
- long double [std::expintl](#) (long double __x)
- template<typename _Tp >
 [__gnu_cxx::__promote](#)< _Tp >::__type [std::hermite](#) (unsigned int __n, _Tp __x)
- float [std::hermitef](#) (unsigned int __n, float __x)
- long double [std::hermitel](#) (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
 [__gnu_cxx::__promote_4](#)< _Tpa, _Tpb, _Tpc, _Tp >::__type [__gnu_cxx::hyperg](#) (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)
- float [__gnu_cxx::hypergf](#) (float __a, float __b, float __c, float __x)
- long double [__gnu_cxx::hypergl](#) (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
 [__gnu_cxx::__promote](#)< _Tp >::__type [std::laguerre](#) (unsigned int __n, _Tp __x)
- float [std::laguerref](#) (unsigned int __n, float __x)
- long double [std::laguerrel](#) (unsigned int __n, long double __x)
- template<typename _Tp >
 [__gnu_cxx::__promote](#)< _Tp >::__type [std::legendre](#) (unsigned int __l, _Tp __x)
- float [std::legendref](#) (unsigned int __l, float __x)
- long double [std::legendrel](#) (unsigned int __l, long double __x)

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::riemann_zeta (_Tp __s)`
- `float std::riemann_zetaf (float __s)`
- `long double std::riemann_zetal (long double __s)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_bessel (unsigned int __n, _Tp __x)`
- `float std::sph_besself (unsigned int __n, float __x)`
- `long double std::sph_bessell (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta)`
- `float std::sph_legendref (unsigned int __l, unsigned int __m, float __theta)`
- `long double std::sph_legendrel (unsigned int __l, unsigned int __m, long double __theta)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_neumann (unsigned int __n, _Tp __x)`
- `float std::sph_neumannf (unsigned int __n, float __x)`
- `long double std::sph_neumannl (unsigned int __n, long double __x)`

7.137.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cmath>`.

7.138 sstream.tcc File Reference

Namespaces

- namespace `std`

Macros

- `#define _SSTREAM_TCC`

7.138.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<sstream>`.

7.139 std_abs.h File Reference

Namespaces

- namespace `std`

Functions

- `constexpr double std::abs (double __x)`
- `constexpr float std::abs (float __x)`
- `long std::abs (long __i)`
- `constexpr long double std::abs (long double __x)`
- `long long std::abs (long long __x)`

7.139.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cmath>` or `<cstdlib>`.

7.140 `std_function.h` File Reference

Classes

- struct [std::__is_location_invariant<_Tp>](#)
- class [std::_Function_base](#)
- class [std::bad_function_call](#)
- class [std::function<_Res\(_ArgTypes...\)>](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Enumerations

- enum [_Manager_operation](#) { [__get_type_info](#) , [__get_functor_ptr](#) , [__clone_functor](#) , [__destroy_functor](#) }

Functions

- template<typename _Functor, typename _Signature = typename __function_guide_helper<decltype(&_Functor::operator())>::type>
std::function (_Functor) -> function<_Signature>
- template<typename _Res, typename... _ArgTypes>
std::function (_Res(*)(_ArgTypes...)) -> function<_Res(_ArgTypes...)>
- template<typename _Res, typename... _Args>
bool [std::operator==](#) (const function<_Res(_Args...)> &__f, nullptr_t) noexcept
- template<typename _Res, typename... _Args>
void [std::swap](#) (function<_Res(_Args...)> &__x, function<_Res(_Args...)> &__y) noexcept

7.140.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.141 `std_mutex.h` File Reference

Classes

- struct [std::adopt_lock_t](#)
- struct [std::defer_lock_t](#)
- class [std::lock_guard<_Mutex>](#)
- class [std::mutex](#)
- struct [std::try_to_lock_t](#)

Namespaces

- namespace [std](#)

Variables

- constexpr [adopt_lock_t](#) [std::adopt_lock](#)
- constexpr [defer_lock_t](#) [std::defer_lock](#)
- constexpr [try_to_lock_t](#) [std::try_to_lock](#)

7.141.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<mutex>`.

7.142 std_thread.h File Reference

Classes

- struct [std::hash< thread::id >](#)
- class [std::thread::id](#)
- class [std::thread](#)

Namespaces

- namespace [std](#)
- namespace [std::this_thread](#)

Functions

- [thread::id std::this_thread::get_id \(\)](#) noexcept
- void [std::this_thread::yield \(\)](#) noexcept

7.142.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<thread>`.

7.143 stl_algo.h File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_clamp`
- `#define __cpp_lib_sample`

Enumerations

- enum { [_S_threshold](#) }
- enum { [_S_chunk_size](#) }

Functions

- template<typename [_ForwardIterator](#) , typename [_BinaryPredicate](#) >
constexpr [_ForwardIterator](#) [std::__adjacent_find](#) ([_ForwardIterator](#) __first, [_ForwardIterator](#) __last, [_BinaryPredicate](#) __binary_pred)
- template<typename [_RandomAccessIterator](#) , typename [_Distance](#) , typename [_Compare](#) >
constexpr void [std::__chunk_insertion_sort](#) ([_RandomAccessIterator](#) __first, [_RandomAccessIterator](#) __last, [_Distance](#) __chunk_size, [_Compare](#) __comp)
- template<typename [_InputIterator](#) , typename [_Size](#) , typename [_OutputIterator](#) >
constexpr [_OutputIterator](#) [std::__copy_n](#) ([_InputIterator](#) __first, [_Size](#) __n, [_OutputIterator](#) __result, [input_iterator_tag](#))

- `template<typename _RandomAccessIterator, typename _Size, typename _OutputIterator >`
`constexpr _OutputIterator std::copy_n (_RandomAccessIterator __first, _Size __n, _OutputIterator __result,`
`random_access_iterator_tag)`
- `template<typename _ForwardIterator, typename _Tp, typename _CompareItTp, typename _CompareTpIt >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _CompareItTp __comp_it_val, _CompareTpIt __comp_val_it)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::final_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BinaryPredicate >`
`constexpr _BidirectionalIterator1 std::find_end (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, bidirectional_iterator_tag, bidirectional_iterator_tag, _BinaryPredicate __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, forward_iterator_tag, forward_iterator_tag, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate, typename _Distance >`
`constexpr _InputIterator std::find_if_not_n (_InputIterator __first, _Distance &__len, _Predicate __pred)`
- `template<typename _EuclideanRingElement >`
`constexpr _EuclideanRingElement std::gcd (_EuclideanRingElement __m, _EuclideanRingElement __n)`
- `template<typename _IntType, typename _UniformRandomBitGenerator >`
`pair< _IntType, _IntType > std::gen_two_uniform_ints (_IntType __b0, _IntType __b1, _UniformRandomBitGenerator &&__g)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::heap_select (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`constexpr bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::inplace_stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`constexpr void std::introsselect (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`constexpr void std::introsort_loop (_RandomAccessIterator __first, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`

- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::__max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::__merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Pointer, typename _Compare >`
`void std::__merge_adaptive (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator1, typename _RandomAccessIterator2, typename _Distance, typename _Compare >`
`void std::__merge_sort_loop (_RandomAccessIterator1 __first, _RandomAccessIterator1 __last, _RandomAccessIterator2 __result, _Distance __step_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Compare >`
`void std::__merge_sort_with_buffer (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Compare >`
`void std::__merge_without_buffer (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::__min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::__minmax_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Iterator, typename _Compare >`
`constexpr void std::__move_median_to_first (_Iterator __result, _Iterator __a, _Iterator __b, _Iterator __c, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__move_merge (_InputIterator __first1, _InputIterator __last1, _InputIterator __first2, _InputIterator __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`void std::__move_merge_adaptive (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BidirectionalIterator3, typename _Compare >`
`void std::__move_merge_adaptive_backward (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, _BidirectionalIterator3 __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::__next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::__partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Predicate >`
`constexpr _BidirectionalIterator std::__partition (_BidirectionalIterator __first, _BidirectionalIterator __last, _Predicate __pred, bidirectional_iterator_tag)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::__partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, forward_iterator_tag)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::__prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`

- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator`
`__result, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`constexpr _OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator`
`__result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _BidirectionalIterator >`
`constexpr void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::reverse (_RandomAccessIterator __first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator >`
`constexpr _BidirectionalIterator std::rotate (_BidirectionalIterator __first, _BidirectionalIterator __middle, ↵`
`_BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator`
`__last, forward_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator std::rotate (_RandomAccessIterator __first, _RandomAccessIterator __↵`
`middle, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _Distance >`
`_BidirectionalIterator1 std::rotate_adaptive (_BidirectionalIterator1 __first, _BidirectionalIterator1 __middle, ↵`
`_BidirectionalIterator1 __last, _Distance __len1, _Distance __len2, _BidirectionalIterator2 __buffer, _Distance`
`__buffer_size)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Cat, typename _Size, typename _UniformRandomBit↵`
`Generator >`
`_OutputIterator std::sample (_ForwardIterator __first, _ForwardIterator __last, forward_iterator_tag, _Output↵`
`Iterator __out, _Cat, _Size __n, _UniformRandomBitGenerator &&__g)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Size, typename _UniformRandomBitGenerator >`
`_RandomAccessIterator std::sample (_InputIterator __first, _InputIterator __last, input_iterator_tag, ↵`
`_RandomAccessIterator __out, random_access_iterator_tag, _Size __n, _UniformRandomBitGenerator &&__g)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _Forward↵`
`Iterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`constexpr _ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count,`
`_UnaryPredicate __unary_pred)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`constexpr _ForwardIterator std::search_n_aux (_ForwardIterator __first, _ForwardIterator __last, _Integer ↵`
`__count, _UnaryPredicate __unary_pred, std::forward_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Integer, typename _UnaryPredicate >`
`constexpr _RandomAccessIter std::search_n_aux (_RandomAccessIter __first, _RandomAccessIter __last, ↵`
`_Integer __count, _UnaryPredicate __unary_pred, std::random_access_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _Input↵`
`Iterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, ↵`
`_InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 ↵`
`__first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::__stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Pointer, typename _Predicate, typename _Distance >`
`_ForwardIterator std::__stable_partition_adaptive (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, _Distance __len, _Pointer __buffer, _Distance __buffer_size)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::__stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Distance, typename _Compare >`
`void std::__stable_sort_adaptive (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__unguarded_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__unguarded_linear_insert (_RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::__unguarded_partition (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __pivot, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::__unguarded_partition_pivot (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::__unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator std::__unique_copy (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred, forward_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::__unique_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _BinaryPredicate __binary_pred, input_iterator_tag, forward_iterator_tag)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator std::__unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred, input_iterator_tag, output_iterator_tag)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator std::__upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`

- `template<typename _Tp >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::clamp (const _Tp &__val, const _Tp &__lo, const _Tp &__hi, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator std::copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _↵ Predicate __pred)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`constexpr _OutputIterator std::copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr iterator_traits< _InputIterator >::difference_type std::count (_InputIterator __first, _InputIterator __last, const _Tp &__value)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr iterator_traits< _InputIterator >::difference_type std::count_if (_InputIterator __first, _InputIterator _↵ __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _Forward↵ Iterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _Forward↵ Iterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _InputIterator std::find (_InputIterator __first, _InputIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _Forward↵ Iterator2 __first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _Forward↵ Iterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`constexpr _InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Function >`
`constexpr _Function std::for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _InputIterator, typename _Size, typename _Function >`
`constexpr _InputIterator std::for_each_n (_InputIterator __first, _Size __n, _Function __f)`
- `template<typename _ForwardIterator, typename _Generator >`
`constexpr void std::generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`constexpr _OutputIterator std::generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵ Iterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`constexpr bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵ Iterator2 __last2, _Compare __comp)`

- `template<typename _BidirectionalIterator >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵
last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵
last, _Compare __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::is_partitioned (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↵
__first2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↵
__first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↵
__first2, _ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator >`
`constexpr bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __↵
__comp)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val,
_Compare __comp)`
- `template<typename _Tp >`
`constexpr _Tp std::max (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr _Tp std::max (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __↵
comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Tp >`
`constexpr _Tp std::min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr _Tp std::min (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __↵
comp)`

- `template<typename _Tp >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`constexpr pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, ↵
_ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`constexpr pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, ↵
_ForwardIterator __last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`constexpr bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare ↵
comp)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool std::none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _Random↵
AccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _Random↵
AccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _Random↵
AccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _Random↵
AccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _Random↵
AccessIterator __result_first, _RandomAccessIterator __result_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _Random↵
AccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`constexpr pair< _OutputIterator1, _OutputIterator2 > std::partition_copy (_InputIterator __first, _InputIterator ↵
__last, _OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate ↵
__pred)`
- `template<typename _BidirectionalIterator >`
`constexpr bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`constexpr bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare ↵
comp)`

- `template<typename _RandomAccessIterator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator &&__rand)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator std::remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`constexpr _OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr _ForwardIterator std::remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator std::replace_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`constexpr _OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`constexpr void std::replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _BidirectionalIterator >`
`constexpr void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, _OutputIterator __result)`
- `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomBitGenerator >`
`_SampleIterator std::sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomBitGenerator &&__g)`
- `template<typename _ForwardIterator, typename _Searcher >`
`constexpr _ForwardIterator std::search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &__searcher)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`constexpr _ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp &__val)`

- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count,`
`const _Tp &__val, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`constexpr _OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`constexpr _OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void std::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator &&__g)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _ForwardIterator >`
`constexpr _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`constexpr _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`

- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`constexpr _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result,`
`_BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵`
`val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`constexpr _ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵`
`val, _Compare __comp)`

7.143.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

7.144 `std_algobase.h` File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_robust_nonmodifying_seq_ops`
- `#define GLIBCXX_MOVE3(_Tp, _Up, _Vp)`
- `#define GLIBCXX_MOVE_BACKWARD3(_Tp, _Up, _Vp)`

Functions

- `template<bool _IsMove, typename _II, typename _OI >`
`constexpr _OI std::copy_move_a (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _II, typename _Ite, typename _Seq, typename _Cat >`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > std::copy_move_a (_II, _II, const __gnu_debug::Safe_iterator<`
`_Ite, _Seq, _Cat > &)`
- `template<bool _IsMove, typename _Ite, typename _ISeq, typename _ICat, typename _Olte, typename _OSeq, typename _OCat >`
`__gnu_debug::Safe_iterator< _Olte, _OSeq, _OCat > std::copy_move_a (const __gnu_debug::Safe_iterator<`
`_Ite, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator< _Ite, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator<`
`_Olte, _OSeq, _OCat > &)`
- `template<bool _IsMove, typename _Ite, typename _Seq, typename _Cat, typename _OI >`
`_OI std::copy_move_a (const __gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &, const __gnu_debug::Safe_iterator<`
`_Ite, _Seq, _Cat > &, _OI)`
- `template<bool _IsMove, typename _ITp, typename _IRef, typename _IPtr, typename _OTp >`
`__Deque_iterator< _OTp, _OTp &, _OTp * > std::copy_move_a1 (__Deque_iterator< _ITp, _IRef, _IPtr >`
`__first, __Deque_iterator< _ITp, _IRef, _IPtr > __last, __Deque_iterator< _OTp, _OTp &, _OTp * > __result)`
- `template<bool _IsMove, typename _Tp, typename _Ref, typename _Ptr, typename _OI >`
`_OI std::copy_move_a1 (__Deque_iterator< _Tp, _Ref, _Ptr > __first, __Deque_iterator< _Tp, _Ref, _Ptr >`
`__last, _OI __result)`
- `template<bool _IsMove, typename _II, typename _Tp >`
`__gnu_cxx::enable_if< __is_random_access_iter< _II >::value, __Deque_iterator< _Tp, _Tp &, _Tp * >`
`>::type std::copy_move_a1 (_II __first, _II __last, __Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<bool _IsMove, typename _II, typename _OI >`
`constexpr _OI std::copy_move_a1 (_II __first, _II __last, _OI __result)`

- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT, char_traits< _CharT >`
`> >::__type std::__copy_move_a2 (_CharT *, _CharT *, ostreambuf_iterator< _CharT, char_traits< _CharT`
`> >)`
- `template<bool _IsMove, typename _II , typename _OI >`
`constexpr _OI std::__copy_move_a2 (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT, char_traits< _CharT`
`> >::__type std::__copy_move_a2 (const _CharT *, const _CharT *, ostreambuf_iterator< _CharT,`
`char_traits< _CharT > >)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value,:: Deque_iterator< _CharT, _CharT &, _CharT`
`* > >::__type std::__copy_move_a2 (istreambuf_iterator< _CharT, char_traits< _CharT > > __first,`
`istreambuf_iterator< _CharT, char_traits< _CharT > > __last, :: Deque_iterator< _CharT, _CharT &, _`
`CharT * > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type std::__copy_move_a2`
`(istreambuf_iterator< _CharT, char_traits< _CharT > >, istreambuf_iterator< _CharT, char_traits< _CharT`
`> >, _CharT *)`
- `template<bool _IsMove, typename _II , typename _OI >`
`constexpr _OI std::__copy_move_backward_a (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _II , typename _Ite , typename _Seq , typename _Cat >`
`__gnu_debug::__Safe_iterator< _Ite, _Seq, _Cat > std::__copy_move_backward_a (_II, _II, const`
`:: __gnu_debug::__Safe_iterator< _Ite, _Seq, _Cat > &)`
- `template<bool _IsMove, typename _Ite , typename _ISeq , typename _ICat , typename _Olte , typename _OSeq , typename _OCat >`
`:: __gnu_debug::__Safe_iterator< _Olte, _OSeq, _OCat > std::__copy_move_backward_a (const :: __gnu_debug::__Safe_iterator<`
`_Ite, _ISeq, _ICat > &, const :: __gnu_debug::__Safe_iterator< _Ite, _ISeq, _ICat > &, const :: __gnu_debug::__Safe_iterator<`
`_Olte, _OSeq, _OCat > &)`
- `template<bool _IsMove, typename _Ite , typename _Seq , typename _Cat , typename _OI >`
`_OI std::__copy_move_backward_a (const :: __gnu_debug::__Safe_iterator< _Ite, _Seq, _Cat > &, const`
`:: __gnu_debug::__Safe_iterator< _Ite, _Seq, _Cat > &, _OI)`
- `template<bool _IsMove, typename _ITp , typename _IRef , typename _IPtr , typename _OTp >`
`:: Deque_iterator< _OTp, _OTp &, _OTp * > std::__copy_move_backward_a1 (:: Deque_iterator< _ITp, _`
`IRef, _IPtr > __first, :: Deque_iterator< _ITp, _IRef, _IPtr > __last, :: Deque_iterator< _OTp, _OTp &, _OTp *`
`> __result)`
- `template<bool _IsMove, typename _Tp , typename _Ref , typename _Ptr , typename _OI >`
`_OI std::__copy_move_backward_a1 (:: Deque_iterator< _Tp, _Ref, _Ptr > __first, :: Deque_iterator< _Tp,`
`_Ref, _Ptr > __last, _OI __result)`
- `template<bool _IsMove, typename _BI1 , typename _BI2 >`
`constexpr _BI2 std::__copy_move_backward_a1 (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<bool _IsMove, typename _II , typename _Tp >`
`__gnu_cxx::__enable_if< __is_random_access_iter< _II >::__value,:: Deque_iterator< _Tp, _Tp &, _Tp * >`
`>::__type std::__copy_move_backward_a1 (_II __first, _II __last, :: Deque_iterator< _Tp, _Tp &, _Tp * >`
`__result)`
- `template<bool _IsMove, typename _BI1 , typename _BI2 >`
`constexpr _BI2 std::__copy_move_backward_a2 (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _InputIterator , typename _Size , typename _OutputIterator >`
`constexpr _OutputIterator std::__copy_n_a (_InputIterator __first, _Size __n, _OutputIterator __result, bool)`
- `template<typename _CharT , typename _Size >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value,:: Deque_iterator< _CharT, _CharT &, _CharT * >`
`>::__type std::__copy_n_a (istreambuf_iterator< _CharT, char_traits< _CharT > > __it, _Size __size, ::`
`Deque_iterator< _CharT, _CharT &, _CharT * > __result, bool __strict)`

- `template<typename _CharT, typename _Size >`
`__gnu_cxx::__enable_if<__is_char<_CharT>::__value, _CharT * >::__type std::__copy_n_a (istreambuf_iterator<_CharT, char_traits<_CharT> >, _Size, _CharT *, bool)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr iterator_traits<_InputIterator>::difference_type std::__count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _II1, typename _II2 >`
`constexpr bool std::__equal4 (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _BinaryPredicate >`
`constexpr bool std::__equal4 (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _BinaryPredicate __binary, _pred)`
- `template<typename _II1, typename _II2 >`
`constexpr bool std::__equal_aux (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _II1, typename _II2, typename _Seq2, typename _Cat2 >`
`bool std::__equal_aux (_II1, _II1, const ::__gnu_debug::Safe_iterator<_II2, _Seq2, _Cat2> &)`
- `template<typename _II1, typename _Seq1, typename _Cat1, typename _II2 >`
`bool std::__equal_aux (const ::__gnu_debug::Safe_iterator<_II1, _Seq1, _Cat1> &, const ::__gnu_debug::Safe_iterator<_II1, _Seq1, _Cat1> &, _II2)`
- `template<typename _II1, typename _Seq1, typename _Cat1, typename _II2, typename _Seq2, typename _Cat2 >`
`bool std::__equal_aux (const ::__gnu_debug::Safe_iterator<_II1, _Seq1, _Cat1> &, const ::__gnu_debug::Safe_iterator<_II1, _Seq1, _Cat1> &, const ::__gnu_debug::Safe_iterator<_II2, _Seq2, _Cat2> &)`
- `template<typename _Tp, typename _Ref, typename _Ptr, typename _II >`
`__gnu_cxx::__enable_if<__is_random_access_iter<_II>::__value, bool >::__type std::__equal_aux1 (::Deque_iterator<_Tp, _Ref, _Ptr> __first1, ::Deque_iterator<_Tp, _Ref, _Ptr> __last1, _II __first2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::__equal_aux1 (::Deque_iterator<_Tp1, _Ref1, _Ptr1> __first1, ::Deque_iterator<_Tp1, _Ref1, _Ptr1> __last1, ::Deque_iterator<_Tp2, _Ref2, _Ptr2> __first2)`
- `template<typename _II, typename _Tp, typename _Ref, typename _Ptr >`
`__gnu_cxx::__enable_if<__is_random_access_iter<_II>::__value, bool >::__type std::__equal_aux1 (_II __first1, _II __last1, ::Deque_iterator<_Tp, _Ref, _Ptr> __first2)`
- `template<typename _II1, typename _II2 >`
`constexpr bool std::__equal_aux1 (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _Flte, typename _Tp >`
`constexpr void std::__fill_a (_Flte __first, _Flte __last, const _Tp & __value)`
- `template<typename _Lte, typename _Seq, typename _Cat, typename _Tp >`
`void std::__fill_a (const ::__gnu_debug::Safe_iterator<_Lte, _Seq, _Cat> &, const ::__gnu_debug::Safe_iterator<_Lte, _Seq, _Cat> &, const _Tp &)`
- `template<typename _Lte, typename _Cont, typename _Tp >`
`constexpr void std::__fill_a1 (::__gnu_cxx::__normal_iterator<_Lte, _Cont> __first, ::__gnu_cxx::__normal_iterator<_Lte, _Cont> __last, const _Tp & __value)`
- `constexpr void std::__fill_a1 (::Bit_iterator, ::Bit_iterator, const bool &)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr __gnu_cxx::__enable_if<!__is_scalar<_Tp>::__value, void >::__type std::__fill_a1 (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __value)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr __gnu_cxx::__enable_if<__is_scalar<_Tp>::__value, void >::__type std::__fill_a1 (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __value)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if<__is_byte<_Tp>::__value, void >::__type std::__fill_a1 (_Tp * __first, _Tp * __last, const _Tp & __c)`
- `template<typename _Tp, typename _VTp >`
`void std::__fill_a1 (const ::Deque_iterator<_Tp, _Tp &, _Tp * > & __first, const ::Deque_iterator<_Tp, _Tp &, _Tp * > & __last, const _VTp & __value)`

- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator std::fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::input_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator std::fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::output_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr _OutputIterator std::fill_n_a (_OutputIterator __first, _Size __n, const _Tp &__value, std::random_access_iterator_tag)`
- `template<typename _Ite, typename _Seq, typename _Cat, typename _Size, typename _Tp >`
`::gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > std::fill_n_a (const ::gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &__first, _Size __n, const _Tp &__value, std::input_iterator_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr __gnu_cxx::enable_if<! is_scalar< _Tp >::value, _OutputIterator >::type std::fill_n_a1 (_OutputIterator __first, _Size __n, const _Tp &__value)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`constexpr __gnu_cxx::enable_if< is_scalar< _Tp >::value, _OutputIterator >::type std::fill_n_a1 (_OutputIterator __first, _Size __n, const _Tp &__value)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr _InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, input_iterator_tag)`
- `template<typename _Iterator, typename _Predicate >`
`constexpr _Iterator std::find_if (_Iterator __first, _Iterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator, typename _Predicate >`
`constexpr _RandomAccessIterator std::find_if (_RandomAccessIterator __first, _RandomAccessIterator __last, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _BinaryPredicate __pred)`
- `template<typename _I1, typename _I2 >`
`constexpr bool std::lexicographical_compare_aux (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _Iter2, typename _Seq2, typename _Cat2 >`
`bool std::lexicographical_compare_aux (_I1, _I1, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, __Cat2 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, __Cat2 > &)`
- `template<typename _Iter1, typename _Seq1, typename _Cat1, typename _I2 >`
`bool std::lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, _I2, _I2)`
- `template<typename _Iter1, typename _Seq1, typename _Cat1, typename _Iter2, typename _Seq2, typename _Cat2 >`
`bool std::lexicographical_compare_aux (const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter1, _Seq1, _Cat1 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &, const ::gnu_debug::Safe_iterator< _Iter2, _Seq2, _Cat2 > &)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::lexicographical_compare_aux1 (::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, ::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`
- `template<typename _Tp1, typename _Ref1, typename _Ptr1, typename _Tp2 >`
`bool std::lexicographical_compare_aux1 (::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __first1, ::Deque_iterator< _Tp1, _Ref1, _Ptr1 > __last1, _Tp2 * __first2, _Tp2 * __last2)`
- `template<typename _I1, typename _I2 >`
`constexpr bool std::lexicographical_compare_aux1 (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _Tp1, typename _Tp2, typename _Ref2, typename _Ptr2 >`
`bool std::lexicographical_compare_aux1 (_Tp1 * __first1, _Tp1 * __last1, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __first2, ::Deque_iterator< _Tp2, _Ref2, _Ptr2 > __last2)`
- `template<typename _I1, typename _I2, typename _Compare >`
`constexpr bool std::lexicographical_compare_impl (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, __Compare __comp)`

- constexpr int `std::__lg` (int __n)
- constexpr long `std::__lg` (long __n)
- constexpr long long `std::__lg` (long long __n)
- constexpr unsigned `std::__lg` (unsigned __n)
- constexpr unsigned long `std::__lg` (unsigned long __n)
- constexpr unsigned long long `std::__lg` (unsigned long long __n)
- template<typename _ForwardIterator, typename _Tp, typename _Compare >
constexpr _ForwardIterator `std::__lower_bound` (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)
- template<typename _Tp, typename _Up >
constexpr int `std::__memcmp` (const _Tp *__first1, const _Up *__first2, size_t __num)
- template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >
constexpr pair<_InputIterator1, _InputIterator2> `std::__mismatch` (_InputIterator1 __first1, _InputIterator1 <- __last1, _InputIterator2 __first2, _BinaryPredicate __binary_pred)
- template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >
constexpr pair<_InputIterator1, _InputIterator2> `std::__mismatch` (_InputIterator1 __first1, _InputIterator1 <- __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)
- template<typename _Iterator >
constexpr _Iterator `std::__niter_base` (_Iterator __it) noexcept(*/*conditional */*)
- template<typename _Ite, typename _Seq >
_Ite `std::__niter_base` (const ::__gnu_debug::Safe_iterator<_Ite, _Seq, std::random_access_iterator_tag> &)
- template<typename _From, typename _To >
constexpr _From `std::__niter_wrap` (_From __from, _To __res)
- template<typename _Iterator >
constexpr _Iterator `std::__niter_wrap` (const _Iterator &, _Iterator __res)
- template<typename _ForwardIterator, typename _Predicate >
constexpr _ForwardIterator `std::__remove_if` (_ForwardIterator __first, _ForwardIterator __last, _Predicate __<- pred)
- constexpr long long `std::__size_to_integer` (double __n)
- constexpr long long `std::__size_to_integer` (float __n)
- constexpr int `std::__size_to_integer` (int __n)
- constexpr long `std::__size_to_integer` (long __n)
- constexpr long long `std::__size_to_integer` (long double __n)
- constexpr long long `std::__size_to_integer` (long long __n)
- constexpr unsigned `std::__size_to_integer` (unsigned __n)
- constexpr unsigned long `std::__size_to_integer` (unsigned long __n)
- constexpr unsigned long long `std::__size_to_integer` (unsigned long long __n)
- template<typename _II, typename _OI >
constexpr _OI `std::copy` (_II __first, _II __last, _OI __result)
- template<typename _BI1, typename _BI2 >
constexpr _BI2 `std::copy_backward` (_BI1 __first, _BI1 __last, _BI2 __result)
- template<typename _II1, typename _II2 >
constexpr bool `std::equal` (_II1 __first1, _II1 __last1, _II2 __first2)
- template<typename _II1, typename _II2 >
constexpr bool `std::equal` (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)
- template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >
constexpr bool `std::equal` (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _BinaryPredicate __binary_pred)
- template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >
constexpr bool `std::equal` (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _IIter2 __last2, _BinaryPredicate __<- binary_pred)

- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _OI, typename _Size, typename _Tp >`
`constexpr _OI std::fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr void std::iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`
- `template<typename _II1, typename _II2 >`
`constexpr bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`constexpr bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr _ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _Tp >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`constexpr const _Tp & std::min (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`constexpr pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _II, typename _OI >`
`constexpr _OI std::move (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`constexpr _BI2 std::move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`constexpr _ForwardIterator2 std::swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`

7.144.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

7.145 `std_bvector.h` File Reference

Classes

- struct [std::hash<::vector< bool, _Alloc > >](#)
- class [std::vector< bool, _Alloc >](#)

Namespaces

- namespace [std](#)

Typedefs

- typedef unsigned long `std::_Bit_type`

Enumerations

- enum { `_S_word_bit` }

Functions

- constexpr void `std::_fill_a1` (::_Bit_iterator, ::_Bit_iterator, const bool &)
- constexpr void `std::_fill_bvector` (_Bit_type * __v, unsigned int __first, unsigned int __last, bool __x) noexcept
- constexpr void `std::_fill_bvector_n` (_Bit_type *, size_t, bool) noexcept

7.145.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

7.146 `std_construct.h` File Reference

Namespaces

- namespace [std](#)

Functions

- template<typename _Tp, typename... _Args>
constexpr void `std::_Construct` (_Tp * __p, _Args &&... __args)
- template<typename _T1 >
void `std::_Construct_novalue` (_T1 * __p)
- template<typename _ForwardIterator >
constexpr void `std::_Destroy` (_ForwardIterator __first, _ForwardIterator __last)
- template<typename _Tp >
constexpr void `std::_Destroy` (_Tp * __pointer)
- template<typename _ForwardIterator, typename _Size >
constexpr _ForwardIterator `std::_Destroy_n` (_ForwardIterator __first, _Size __count)
- template<typename _Tp, typename... _Args>
constexpr auto `std::construct_at` (_Tp * __location, _Args &&... __args) noexcept(noexcept(::new((void *) 0) _Tp(std::declval< _Args >()...))) -> decltype(::new((void *) 0) _Tp(std::declval< _Args >()...))
- template<typename _ForwardIterator >
constexpr void `std::destroy` (_ForwardIterator __first, _ForwardIterator __last)
- template<typename _Tp >
constexpr void `std::destroy_at` (_Tp * __location)
- template<typename _ForwardIterator, typename _Size >
constexpr _ForwardIterator `std::destroy_n` (_ForwardIterator __first, _Size __count)

7.146.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.147 std_deque.h File Reference

Classes

- class [std::_Deque_base<_Tp, _Alloc>](#)
- struct [std::_Deque_iterator<_Tp, _Ref, _Ptr>](#)
- class [std::deque<_Tp, _Alloc>](#)

Namespaces

- namespace [std](#)

Macros

- [#define _GLIBCXX_DEQUE_BUF_SIZE](#)

Functions

- constexpr size_t [std::__deque_buf_size](#) (size_t __size)
- template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>
[std::deque](#) (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> [deque](#)<_ValT, _Allocator>
- template<typename _Tp, typename _Alloc>
bool [std::operator!=](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
bool [std::operator<](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
bool [std::operator<=](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
bool [std::operator==](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
bool [std::operator>](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
bool [std::operator>=](#) (const [deque](#)<_Tp, _Alloc> &__x, const [deque](#)<_Tp, _Alloc> &__y)
- template<typename _Tp, typename _Alloc>
void [std::swap](#) ([deque](#)<_Tp, _Alloc> &__x, [deque](#)<_Tp, _Alloc> &__y) noexcept(*/*conditional */*)

7.147.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<deque>`.

7.147.2 Macro Definition Documentation

[_GLIBCXX_DEQUE_BUF_SIZE](#)

```
#define _GLIBCXX_DEQUE_BUF_SIZE
```

This function controls the size of memory nodes.

Parameters

<code>__size</code>	The size of an element.
---------------------	-------------------------

Returns

The number (not byte size) of elements per node.

This function started off as a compiler kludge from SGI, but seems to be a useful wrapper around a repeated constant expression. The **512** is tunable (and no other code needs to change), but no investigation has been done since inheriting the SGI code. Touch `_GLIBCXX_DEQUE_BUF_SIZE` only if you know what you are doing, however: changing it breaks the binary compatibility!!

7.148 `std_function.h` File Reference

Classes

- struct `std::binary_function<_Arg1, _Arg2, _Result >`
- class `std::binary_negate<_Predicate >`
- class `std::const_mem_fun1_ref_t<_Ret, _Tp, _Arg >`
- class `std::const_mem_fun1_t<_Ret, _Tp, _Arg >`
- class `std::const_mem_fun_ref_t<_Ret, _Tp >`
- class `std::const_mem_fun_t<_Ret, _Tp >`
- struct `std::divides<_Tp >`
- struct `std::divides<void >`
- struct `std::equal_to<_Tp >`
- struct `std::equal_to<void >`
- struct `std::greater<_Tp >`
- struct `std::greater<void >`
- struct `std::greater_equal<_Tp >`
- struct `std::greater_equal<void >`
- struct `std::less<_Tp >`
- struct `std::less<void >`
- struct `std::less_equal<_Tp >`
- struct `std::less_equal<void >`
- struct `std::logical_and<_Tp >`
- struct `std::logical_and<void >`
- struct `std::logical_not<_Tp >`
- struct `std::logical_not<void >`
- struct `std::logical_or<_Tp >`
- struct `std::logical_or<void >`
- class `std::mem_fun1_ref_t<_Ret, _Tp, _Arg >`
- class `std::mem_fun1_t<_Ret, _Tp, _Arg >`
- class `std::mem_fun_ref_t<_Ret, _Tp >`
- class `std::mem_fun_t<_Ret, _Tp >`
- struct `std::minus<_Tp >`
- struct `std::minus<void >`
- struct `std::modulus<_Tp >`
- struct `std::modulus<void >`
- struct `std::multiplies<_Tp >`
- struct `std::multiplies<void >`
- struct `std::negate<_Tp >`
- struct `std::negate<void >`
- struct `std::not_equal_to<_Tp >`
- struct `std::not_equal_to<void >`
- struct `std::plus<_Tp >`
- class `std::pointer_to_binary_function<_Arg1, _Arg2, _Result >`
- class `std::pointer_to_unary_function<_Arg, _Result >`
- struct `std::unary_function<_Arg, _Result >`
- class `std::unary_negate<_Predicate >`

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_transparent_operators`

Typedefs

- `template<typename _Func, typename _SfinaeType >`
`using std::__has_is_transparent_t = typename __has_is_transparent< _Func, _SfinaeType >::type`

Functions

- `template<typename _Ret, typename _Tp >`
`const_mem_fun_t< _Ret, _Tp > std::mem_fun (_Ret(_Tp::*__f)() const)`
- `template<typename _Ret, typename _Tp >`
`mem_fun_t< _Ret, _Tp > std::mem_fun (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`const_mem_fun1_t< _Ret, _Tp, _Arg > std::mem_fun (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`mem_fun1_t< _Ret, _Tp, _Arg > std::mem_fun (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Ret, typename _Tp >`
`const_mem_fun_ref_t< _Ret, _Tp > std::mem_fun_ref (_Ret(_Tp::*__f)() const)`
- `template<typename _Ret, typename _Tp >`
`mem_fun_ref_t< _Ret, _Tp > std::mem_fun_ref (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`const_mem_fun1_ref_t< _Ret, _Tp, _Arg > std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`mem_fun1_ref_t< _Ret, _Tp, _Arg > std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Predicate >`
`constexpr unary_negate< _Predicate > std::not1 (const _Predicate &__pred)`
- `template<typename _Predicate >`
`constexpr binary_negate< _Predicate > std::not2 (const _Predicate &__pred)`
- `template<typename _Arg, typename _Result >`
`pointer_to_unary_function< _Arg, _Result > std::ptr_fun (_Result(*__x)(_Arg))`
- `template<typename _Arg1, typename _Arg2, typename _Result >`
`pointer_to_binary_function< _Arg1, _Arg2, _Result > std::ptr_fun (_Result(*__x)(_Arg1, _Arg2))`

7.148.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

7.149 `std_heap.h` File Reference

Namespaces

- namespace [std](#)

Functions

- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`constexpr void std::__adjust_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __len,`
`_Tp __value, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare, typename _Distance >`
`constexpr bool std::__is_heap (_RandomAccessIterator __first, _Compare __comp, _Distance __n)`
- `template<typename _RandomAccessIterator, typename _Distance >`
`constexpr bool std::__is_heap (_RandomAccessIterator __first, _Distance __n)`
- `template<typename _RandomAccessIterator >`
`constexpr bool std::__is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr bool std::__is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`constexpr _Distance std::__is_heap_until (_RandomAccessIterator __first, _Distance __n, _Compare &__comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare`
`&__comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __result,`
`_Compare &__comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`constexpr void std::__push_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __topIndex,`
`_Tp __value, _Compare &__comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::__sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare`
`&__comp)`
- `template<typename _RandomAccessIterator >`
`constexpr bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr _RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr _RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last,`
`_Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`constexpr void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

- `template<typename _RandomAccessIterator >`
`constexpr void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`constexpr void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵
comp)`

7.149.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<queue>`.

7.150 `std_iterator.h` File Reference

Classes

- class [std::back_insert_iterator](#)< _Container >
- class [std::common_iterator](#)< _It, _Sent >
- class [std::counted_iterator](#)< _It >
- class [std::front_insert_iterator](#)< _Container >
- class [std::insert_iterator](#)< _Container >
- class [std::move_iterator](#)< _Iterator >
- class [std::reverse_iterator](#)< _Iterator >

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_array_constexpr`
- `#define __cpp_lib_constexpr_iterator`
- `#define __cpp_lib_make_reverse_iterator`
- `#define __cpp_lib_move_iterator_concept`
- `#define _GLIBCXX_MAKE_MOVE_IF_NOEXCEPT_ITERATOR(_Iter)`
- `#define _GLIBCXX_MAKE_MOVE_ITERATOR(_Iter)`

Typedefs

- `template<typename _Cat, typename _Limit, typename _Otherwise = _Cat>`
`using std::__detail::__clamp_iter_cat = __conditional_t< derived_from< _Cat, _Limit >, _Limit, _Otherwise >`
- `template<typename _InputIterator >`
`using std::__iter_key_t = remove_const_t< typename iterator_traits< _InputIterator >::value_type::first_type >`
- `template<typename _InputIterator >`
`using std::__iter_to_alloc_t = pair< add_const_t< __iter_key_t< _InputIterator > >, __iter_val_t< _Input↵
Iterator > >`
- `template<typename _InputIterator >`
`using std::__iter_val_t = typename iterator_traits< _InputIterator >::value_type::second_type`

Functions

- `template<typename _Iterator, typename _ReturnType = __conditional_t<__move_if_noexcept_cond<typename iterator_traits<_Iterator>::value_type>::value, _Iterator, move_iterator<_Iterator>>>>`
`constexpr _ReturnType std::make_move_if_noexcept_iterator (_Iterator __i)`
- `template<typename _Tp, typename _ReturnType = __conditional_t<__move_if_noexcept_cond<_Tp>::value, const _Tp*, move_iterator<_Tp*>>>>`
`constexpr _ReturnType std::make_move_if_noexcept_iterator (_Tp * __i)`
- `template<typename _Iterator >`
`constexpr reverse_iterator<_Iterator> std::make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`constexpr auto std::__miter_base (move_iterator<_Iterator> __it) -> decltype(__miter_base(__it.base()))`
- `template<typename _Iterator >`
`constexpr auto std::__miter_base (reverse_iterator<_Iterator> __it) -> decltype(__make_reverse_iterator(__miter_base(__it.base())))`
- `template<typename _Iterator, typename _Container >`
`constexpr _Iterator std::niter_base (_gnu_cxx::__normal_iterator<_Iterator, _Container> __it) noexcept(/*conditional */)`
- `template<typename _Iterator >`
`constexpr auto std::__niter_base (move_iterator<_Iterator> __it) -> decltype(make_move_iterator(__niter_base(__it.base())))`
- `template<typename _Iterator >`
`constexpr auto std::__niter_base (reverse_iterator<_Iterator> __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- `template<typename _Container >`
`constexpr back_insert_iterator<_Container> std::back_inserter (_Container & __x)`
- `template<typename _Container >`
`constexpr front_insert_iterator<_Container> std::front_inserter (_Container & __x)`
- `template<typename _Container >`
`constexpr insert_iterator<_Container> std::inserter (_Container & __x, std::detail::__range_iter_t<_Container> __i)`
- `template<typename _Iterator >`
`constexpr move_iterator<_Iterator> std::make_move_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`constexpr reverse_iterator<_Iterator> std::make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator!= (const __normal_iterator<_Iterator, _Container> & __lhs, const __normal_iterator<_Iterator, _Container> & __rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr bool __gnu_cxx::operator!= (const __normal_iterator<_IteratorL, _Container> & __lhs, const __normal_iterator<_IteratorR, _Container> & __rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator!= (const move_iterator<_Iterator> & __x, const move_iterator<_Iterator> & __y)`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr bool std::operator!= (const move_iterator<_IteratorL> & __x, const move_iterator<_IteratorR> & __y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() != __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator!= (const reverse_iterator<_IteratorL> & __x, const reverse_iterator<_IteratorR> & __y)`
- `template<typename _Iterator, typename _Container >`
`constexpr __normal_iterator<_Iterator, _Container> __gnu_cxx::operator+ (typename __normal_iterator<_Iterator, _Container>::difference_type __n, const __normal_iterator<_Iterator, _Container> & __i) noexcept`

- `template<typename _Iterator >`
`constexpr move_iterator< _Iterator > std::operator+ (typename move_iterator< _Iterator >::difference_type __n, const move_iterator< _Iterator > &__x)`
- `template<typename _Iterator >`
`constexpr reverse_iterator< _Iterator > std::operator+ (typename reverse_iterator< _Iterator >::difference_type __n, const reverse_iterator< _Iterator > &__x)`
- `template<typename _Iterator, typename _Container >`
`constexpr __normal_iterator< _Iterator, _Container >::difference_type __gnu_cxx::operator- (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr auto __gnu_cxx::operator- (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept -> decltype(__lhs.base() - __rhs.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto std::operator- (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y) -> decltype(__x.base() - __y.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`constexpr auto std::operator- (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y) -> decltype(__y.base() - __x.base())`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator< (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator< (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator< (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator< (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() > __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator< (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator<= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator<= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator<= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator<= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() >= __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator<= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`

- `template<three_way_comparable _Iterator>`
`constexpr compare_three_way_result_t< _Iterator, _Iterator > std::operator<=> (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, three_way_comparable_with< _IteratorL > _IteratorR>`
`constexpr compare_three_way_result_t< _IteratorL, _IteratorR > std::operator<=> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator== (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`constexpr bool __gnu_cxx::operator== (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator== (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() == __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator== (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator> (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator> (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __y.base() < __x.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`
`constexpr bool std::operator> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Iterator, typename _Container >`
`constexpr bool __gnu_cxx::operator>= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator>= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator >`
`constexpr bool std::operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`requires requires { { __x.base() < __y.base() } -> convertible_to<bool>; }`

```
constexpr bool std::operator>= (const move\_iterator< _IteratorL > &__x, const move\_iterator< _IteratorR >
&__y)
• template<typename _IteratorL , typename _IteratorR >
  requires requires { { __x.base() <= __y.base() } -> convertible_to<bool>; }
constexpr bool std::operator>= (const reverse\_iterator< _IteratorL > &__x, const reverse\_iterator< _IteratorR
> &__y)
```

Variables

- template<typename _Iterator1 , typename _Iterator2 >
 constexpr bool **std::disable_sized_sentinel_for**< [reverse_iterator](#)< _Iterator1 >, [reverse_iterator](#)< _Iterator2 > >

7.150.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file implements `reverse_iterator`, `back_insert_iterator`, `front_insert_iterator`, `insert_iterator`, `__normal_iterator`, and their supporting functions and overloaded operators.

7.151 `std_iterator.h` File Reference

Namespaces

- namespace [__gnu_debug](#)

Functions

- template<typename _Iterator >
 constexpr auto [__gnu_debug::__base](#) (const [std::move_iterator](#)< _Iterator > &__it) -> decltype(std::make_↵
[move_iterator](#)(__base(__it.base())))
- template<typename _Iterator , typename _Sequence >
 constexpr [std::reverse_iterator](#)< _Iterator > [__gnu_debug::__base](#) (const [std::reverse_iterator](#)< [_Safe_iterator](#)<
 _Iterator, _Sequence, [std::random_access_iterator_tag](#) > > &__it)
- template<typename _Iterator , typename _Size >
 constexpr bool [__gnu_debug::__can_advance](#) (const [std::move_iterator](#)< _Iterator > &__it, _Size __n)
- template<typename _Iterator , typename _Diff >
 constexpr bool [__gnu_debug::__can_advance](#) (const [std::move_iterator](#)< _Iterator > &__it, const [std::pair](#)<
 _Diff, [_Distance_precision](#) > &__dist, int __way)
- template<typename _Iterator , typename _Size >
 constexpr bool [__gnu_debug::__can_advance](#) (const [std::reverse_iterator](#)< _Iterator > &__it, _Size __n)
- template<typename _Iterator , typename _Diff >
 constexpr bool [__gnu_debug::__can_advance](#) (const [std::reverse_iterator](#)< _Iterator > &__it, const [std::pair](#)<
 _Diff, [_Distance_precision](#) > &__dist, int __way)
- template<typename _Iterator >
 constexpr [_Distance_traits](#)< _Iterator >::__type [__gnu_debug::__get_distance](#) (const [std::move_iterator](#)< ↵
 _Iterator > &__first, const [std::move_iterator](#)< _Iterator > &__last)
- template<typename _Iterator >
 constexpr [_Distance_traits](#)< _Iterator >::__type [__gnu_debug::__get_distance](#) (const [std::reverse_iterator](#)<
 _Iterator > &__first, const [std::reverse_iterator](#)< _Iterator > &__last)
- template<typename _Iterator >
 constexpr auto [__gnu_debug::__unsafe](#) (const [std::move_iterator](#)< _Iterator > &__it) -> decltype(std::make_↵
 _move_iterator(__unsafe(__it.base())))

- `template<typename _Iterator >`
`constexpr auto __gnu_debug::__unsafe (const std::reverse_iterator< _Iterator > &__it) -> decltype(std::__↵`
`make_reverse_iterator(__unsafe(__it.base())))`
- `template<typename _Iterator >`
`constexpr bool __gnu_debug::__valid_range (const std::move_iterator< _Iterator > &__first, const`
`std::move_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _Iterator >`
`constexpr bool __gnu_debug::__valid_range (const std::reverse_iterator< _Iterator > &__first, const`
`std::reverse_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`

7.151.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.152 `stl_iterator_base_funcs.h` File Reference

Namespaces

- namespace [std](#)

Functions

- `template<typename _BidirectionalIterator, typename _Distance >`
`constexpr void std::__advance (_BidirectionalIterator &__i, _Distance __n, bidirectional_iterator_tag)`
- `template<typename _InputIterator, typename _Distance >`
`constexpr void std::__advance (_InputIterator &__i, _Distance __n, input_iterator_tag)`
- `template<typename _OutputIterator, typename _Distance >`
`void std::__advance (_OutputIterator &, _Distance, output_iterator_tag)=delete`
- `template<typename _RandomAccessIterator, typename _Distance >`
`constexpr void std::__advance (_RandomAccessIterator &__i, _Distance __n, random_access_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::difference_type std::__distance (_InputIterator __first, _InputIterator`
`__last, input_iterator_tag)`
- `template<typename _OutputIterator >`
`void std::__distance (_OutputIterator, _OutputIterator, output_iterator_tag)=delete`
- `template<typename _RandomAccessIterator >`
`constexpr iterator_traits< _RandomAccessIterator >::difference_type std::__distance (_RandomAccessIterator`
`__first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _InputIterator, typename _Distance >`
`constexpr void std::advance (_InputIterator &__i, _Distance __n)`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::difference_type std::distance (_InputIterator __first, _InputIterator __↵`
`__last)`
- `template<typename _InputIterator >`
`constexpr _InputIterator std::next (_InputIterator __x, typename iterator_traits< _InputIterator >::difference_type`
`__n=1)`
- `template<typename _BidirectionalIterator >`
`constexpr _BidirectionalIterator std::prev (_BidirectionalIterator __x, typename iterator_traits< _Bidirectional↵`
`Iterator >::difference_type __n=1)`

7.152.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file contains all of the general iterator-related utility functions, such as `distance()` and `advance()`.

7.153 `std_iterator_base_types.h` File Reference

Classes

- struct [std::bidirectional_iterator_tag](#)
- struct [std::contiguous_iterator_tag](#)
- struct [std::forward_iterator_tag](#)
- struct [std::input_iterator_tag](#)
- struct [std::iterator<_Category, _Tp, _Distance, _Pointer, _Reference>](#)
- struct [std::iterator_traits<_Iterator>](#)
- struct [std::iterator_traits<_Tp*>](#)
- struct [std::output_iterator_tag](#)
- struct [std::random_access_iterator_tag](#)

Namespaces

- namespace [std](#)

Typedefs

- `template<typename _Iter>`
using [std::__iterator_category_t](#) = typename [iterator_traits<_Iter>::iterator_category](#)
- `template<typename _InIter>`
using [std::RequireInputIter](#) = `__enable_if_t<is_convertible<__iterator_category_t<_InIter>, input_iterator_tag`
`>::value>`

Functions

- `template<typename _Iter>`
`constexpr iterator_traits<_Iter>::iterator_category std::__iterator_category \(const _Iter &\)`

7.153.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file contains all of the general iterator-related utility types, such as `iterator_traits` and `struct iterator`.

7.154 `std_list.h` File Reference

Classes

- class [std::_List_base<_Tp, _Alloc>](#)
- struct [std::_List_const_iterator<_Tp>](#)
- struct [std::_List_iterator<_Tp>](#)
- struct [std::_List_node<_Tp>](#)
- struct [std::__detail::_List_node_base](#)
- struct [std::__detail::_List_node_header](#)
- class [std::list<_Tp, _Alloc>](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_list_remove_return_type`
- `#define _GLIBCXX_LIST_REMOVE_RETURN_TYPE_TAG`

Functions

- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::list(_InputIterator, _InputIterator, _Allocator=_Allocator()) -> list<_ValT, _Allocator>`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator!= (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator< (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator<= (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator== (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator> (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`bool std::operator>= (const list<_Tp, _Alloc> &__x, const list<_Tp, _Alloc> &__y)`
- `template<typename _Tp, typename _Alloc>`
`void std::swap(list<_Tp, _Alloc> &__x, list<_Tp, _Alloc> &__y) noexcept(/*conditional */)`

7.154.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<list>`.

7.155 `std_map.h` File Reference

Classes

- class `std::map<_Key, _Tp, _Compare, _Alloc>`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_map_try_emplace`

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::map(_InputIterator, _InputIterator, _Allocator) -> map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, less<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_val_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::map(_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Compare, _Allocator>`

- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::map (initializer_list< pair< _Key, _Tp > >, _Allocator) -> map< _Key, _Tp, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::map (initializer_list< pair< _Key, _Tp > >, _Compare=_Compare(), _Allocator=_Allocator()) -> map< _Key, _Tp, _Compare, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator!= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator< (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator<= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator== (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator> (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator>= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void std::swap (map< _Key, _Tp, _Compare, _Alloc > &__x, map< _Key, _Tp, _Compare, _Alloc > &__y)`
`noexcept(!*conditional */)`

7.155.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>`.

7.156 std_multimap.h File Reference

Classes

- class `std::multimap< _Key, _Tp, _Compare, _Alloc >`

Namespaces

- namespace `std`

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::multimap (_InputIterator, _InputIterator, _Allocator) -> multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, less< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::multimap (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Compare, _Allocator >`

- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::multimap (initializer_list< pair< _Key, _Tp > >, _Allocator) -> multimap< _Key, _Tp, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::multimap (initializer_list< pair< _Key, _Tp > >, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap< _Key, _Tp, _Compare, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator!= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator< (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator<= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator== (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator> (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool std::operator>= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void std::swap (multimap< _Key, _Tp, _Compare, _Alloc > &__x, multimap< _Key, _Tp, _Compare, _Alloc > &__y) noexcept(/*conditional */)`

7.156.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>`.

7.157 `std_multiset.h` File Reference

Classes

- class `std::multiset< _Key, _Compare, _Alloc >`

Namespaces

- namespace `std`

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::multiset (_InputIterator, _InputIterator, _Allocator) -> multiset< typename iterator_traits<_InputIterator>::value_type, less< typename iterator_traits<_InputIterator>::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::multiset (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> multiset< typename iterator_traits<_InputIterator>::value_type, _Compare, _Allocator >`

- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::multiset (initializer_list< _Key >, _Allocator) -> multiset< _Key, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>>`
`std::multiset (initializer_list< _Key >, _Compare=_Compare(), _Allocator=_Allocator()) -> multiset< _Key, _Compare, _Allocator >`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator!= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator< (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator<= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator== (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator> (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool std::operator>= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void std::swap (multiset< _Key, _Compare, _Alloc > &__x, multiset< _Key, _Compare, _Alloc > &__y) noexcept(*conditional *)`

7.157.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<set>`.

7.158 std_numeric.h File Reference

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_MOVE_IF_20(_E)`

Functions

- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`

- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp std::inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2 >`
`constexpr _Tp std::inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`
- `template<typename _ForwardIterator, typename _Tp >`
`constexpr void std::iota (_ForwardIterator __first, _ForwardIterator __last, _Tp __value)`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op)`

7.158.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<numeric>`.

7.159 `std::pair.h` File Reference

Classes

- struct [std::pair](#)< `_T1`, `_T2` >
- struct [std::piecewise_construct_t](#)
- struct [std::tuple_element](#)< 0, [pair](#)< `_Tp1`, `_Tp2` > >
- struct [std::tuple_element](#)< 1, [pair](#)< `_Tp1`, `_Tp2` > >
- struct [std::tuple_size](#)< [pair](#)< `_Tp1`, `_Tp2` > >

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_constexpr_utility`
- `#define _GLIBCXX_DEPRECATED_PAIR_CTOR`

Variables

- `template<typename _Tp >`
`constexpr bool std::__is_pair`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::__is_pair< const pair< _Tp, _Up > >`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::__is_pair< pair< _Tp, _Up > >`
- `constexpr piecewise_construct_t std::piecewise_construct`
- `template<typename _Tp1, typename _Tp2 >`
`constexpr size_t std::tuple_size_v< const pair< _Tp1, _Tp2 > >`

- `template<typename _Tp1, typename _Tp2 >`
`constexpr size_t std::tuple_size_v< pair< _Tp1, _Tp2 > >`
- `#define __cpp_lib_tuples_by_type`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp && std::get (const pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & std::get (const pair< _Tp, _Up > &__p) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && std::get (const pair< _Tp1, _Tp2 > &&__in) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr const tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & std::get (const pair< _Tp1, _Tp2 > &__in) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp && std::get (const pair< _Up, _Tp > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & std::get (const pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && std::get (pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & std::get (pair< _Tp, _Up > &__p) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type && std::get (pair< _Tp1, _Tp2 > &&__in) noexcept`
- `template<size_t _Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< _Int, pair< _Tp1, _Tp2 > >::type & std::get (pair< _Tp1, _Tp2 > &__in) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && std::get (pair< _Up, _Tp > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & std::get (pair< _Up, _Tp > &__p) noexcept`

7.159.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

7.159.2 Macro Definition Documentation

`__cpp_lib_tuples_by_type`

```
#define __cpp_lib_tuples_by_type
```

`std::get` overloads for accessing members of `std::pair`

7.160 `std::queue.h` File Reference

Classes

- class `std::priority_queue< _Tp, _Sequence, _Compare >`
- class `std::queue< _Tp, _Sequence >`

Namespaces

- namespace `std`

Functions

- `template<typename _Tp, typename _Seq>`
`bool std::operator!= (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator< (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator<= (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator== (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator> (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator>= (const queue< _Tp, _Seq> &__x, const queue< _Tp, _Seq> &__y)`
- `template<typename _Compare, typename _Container, typename = _RequireNotAllocator<_Compare>, typename = _RequireNotAllocator<_Container>>>`
`std::priority_queue (_Compare, _Container) -> priority_queue< typename _Container::value_type, _Container, _Compare>`
- `template<typename _Compare, typename _Container, typename _Allocator, typename = _RequireNotAllocator<_Compare>, typename = _RequireNotAllocator<_Container>>>`
`std::priority_queue (_Compare, _Container, _Allocator) -> priority_queue< typename _Container::value_type, _Container, _Compare>`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Compare = less<_ValT>, typename _Container = vector<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireNotAllocator<_Container>>>`
`std::priority_queue (_InputIterator, _InputIterator, _Compare=_Compare(), _Container=_Container()) -> priority_queue< _ValT, _Container, _Compare>`
- `template<typename _Container, typename = _RequireNotAllocator<_Container>>>`
`std::queue (_Container) -> queue< typename _Container::value_type, _Container>`
- `template<typename _Container, typename _Allocator, typename = _RequireNotAllocator<_Container>>>`
`std::queue (_Container, _Allocator) -> queue< typename _Container::value_type, _Container>`
- `template<typename _Tp, typename _Sequence, typename _Compare>`
`enable_if< __and< __is_swappable< _Sequence>, __is_swappable< _Compare>>::value>::type std::swap (priority_queue< _Tp, _Sequence, _Compare> &__x, priority_queue< _Tp, _Sequence, _Compare> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Seq>`
`enable_if< __is_swappable< _Seq>::value>::type std::swap (queue< _Tp, _Seq> &__x, queue< _Tp, _Seq> &__y) noexcept(noexcept(__x.swap(__y)))`

7.160.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<queue>`.

7.161 `std_raw_storage_iter.h` File Reference

Classes

- class `std::raw_storage_iterator< _OutputIterator, _Tp>`

Namespaces

- namespace `std`

7.161.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.162 `std_rel_ops.h` File Reference

Namespaces

- namespace `std`
- namespace `std::rel_ops`

Functions

- `template<class _Tp >`
`bool std::rel_ops::operator!= (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator<= (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator> (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator>= (const _Tp &__x, const _Tp &__y)`

7.162.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

This file is only included by `<utility>`, which is required by the standard to define namespace `rel_ops` and its contents.

7.163 `std_set.h` File Reference

Classes

- class `std::set< _Key, _Compare, _Alloc >`

Namespaces

- namespace `std`

Functions

- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator!= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator< (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator<= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator== (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator> (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator>= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::set` (`_InputIterator`, `_InputIterator`, `_Allocator`) -> `set` `< typename iterator_traits<_InputIterator>::value_type, less< typename iterator_traits<_InputIterator>::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::set` (`_InputIterator`, `_InputIterator`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `set` `< typename iterator_traits<_InputIterator>::value_type, _Compare, _Allocator >`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::set` (`initializer_list<_Key>`, `_Allocator`) -> `set` `< _Key, less<_Key>, _Allocator >`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::set` (`initializer_list<_Key>`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `set` `< _Key, _Compare, _Allocator >`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void std::swap` (`set<_Key, _Compare, _Alloc> &__x`, `set<_Key, _Compare, _Alloc> &__y`) `noexcept` (`/*conditional */`)

7.163.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<set>`.

7.164 `std_stack.h` File Reference

Classes

- class `std::stack<_Tp, _Sequence>`

Namespaces

- namespace `std`

Functions

- `template<typename _Tp, typename _Seq >`
`bool std::operator!=` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Tp, typename _Seq >`
`bool std::operator<` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Tp, typename _Seq >`
`bool std::operator<=` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Tp, typename _Seq >`
`bool std::operator==` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Tp, typename _Seq >`
`bool std::operator>` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Tp, typename _Seq >`
`bool std::operator>=` (`const stack<_Tp, _Seq> &__x`, `const stack<_Tp, _Seq> &__y`)
- `template<typename _Container, typename = _RequireNotAllocator<_Container>>`
`std::stack` (`_Container`) -> `stack` `< typename _Container::value_type, _Container >`
- `template<typename _Container, typename _Allocator, typename = _RequireNotAllocator<_Container>>`
`std::stack` (`_Container`, `_Allocator`) -> `stack` `< typename _Container::value_type, _Container >`
- `template<typename _Tp, typename _Seq >`
`enable_if<_is_swappable<_Seq>::value >::type std::swap` (`stack<_Tp, _Seq> &__x`, `stack<_Tp, _Seq> &__y`) `noexcept` (`noexcept(__x.swap(__y))`)

7.164.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<stack>`.

7.165 `std_tempbuf.h` File Reference

Classes

- class `std::_Temporary_buffer<_ForwardIterator, _Tp>`

Namespaces

- namespace `std`
- namespace `std::__detail`

Functions

- template<typename _Tp>
void `std::__detail::__return_temporary_buffer` (_Tp * __p, size_t __len)
- template<typename _Pointer, typename _ForwardIterator>
void `std::__uninitialized_construct_buf` (_Pointer __first, _Pointer __last, _ForwardIterator __seed)
- template<typename _Tp>
`pair`<_Tp *, ptrdiff_t> `std::get_temporary_buffer` (ptrdiff_t __len) noexcept
- template<typename _Tp>
void `std::return_temporary_buffer` (_Tp * __p)

7.165.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.166 `std_tree.h` File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_generic_associative_lookup`

Enumerations

- enum `_Rb_tree_color` { `_S_red`, `_S_black` }

Functions

- unsigned int `std::_Rb_tree_black_count` (const _Rb_tree_node_base * __node, const _Rb_tree_node_base * __root) throw ()
- _Rb_tree_node_base * `std::_Rb_tree_decrement` (_Rb_tree_node_base * __x) throw ()
- const _Rb_tree_node_base * `std::_Rb_tree_decrement` (const _Rb_tree_node_base * __x) throw ()
- _Rb_tree_node_base * `std::_Rb_tree_increment` (_Rb_tree_node_base * __x) throw ()
- const _Rb_tree_node_base * `std::_Rb_tree_increment` (const _Rb_tree_node_base * __x) throw ()
- void `std::_Rb_tree_insert_and_rebalance` (const bool __insert_left, _Rb_tree_node_base * __x, _Rb_tree_node_base * __p, _Rb_tree_node_base & __header) throw ()

- `_Rb_tree_node_base * std::Rb_tree_rebalance_for_erase` (`_Rb_tree_node_base *const __z, _Rb_tree_node_base &__header`) `throw ()`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`void std::swap` (`_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y`)

7.166.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>` or `<set>`.

7.167 `std_uninitialized.h` File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_raw_memory_algorithms`

Functions

- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_copy` (`_InputIterator __first, _InputIterator __last, _ForwardIterator __result`)
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_copy_n` (`_InputIterator __first, _Size __n, _ForwardIterator __result`)
- `template<typename _ForwardIterator >`
`void std::uninitialized_default_construct` (`_ForwardIterator __first, _ForwardIterator __last`)
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator std::uninitialized_default_construct_n` (`_ForwardIterator __first, _Size __count`)
- `template<typename _ForwardIterator, typename _Tp >`
`void std::uninitialized_fill` (`_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x`)
- `template<typename _ForwardIterator, typename _Size, typename _Tp >`
`_ForwardIterator std::uninitialized_fill_n` (`_ForwardIterator __first, _Size __n, const _Tp &__x`)
- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_move` (`_InputIterator __first, _InputIterator __last, _ForwardIterator __result`)
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`pair< _InputIterator, _ForwardIterator > std::uninitialized_move_n` (`_InputIterator __first, _Size __count, _ForwardIterator __result`)
- `template<typename _ForwardIterator >`
`void std::uninitialized_value_construct` (`_ForwardIterator __first, _ForwardIterator __last`)
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator std::uninitialized_value_construct_n` (`_ForwardIterator __first, _Size __count`)

7.167.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.168 `std_vector.h` File Reference

Classes

- struct `std::_Vector_base< _Tp, _Alloc >`
- class `std::vector< _Tp, _Alloc >`

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_constexpr_vector`
- `#define _GLIBCXX_ASAN_ANNOTATE_BEFORE_DEALLOC`
- `#define _GLIBCXX_ASAN_ANNOTATE_GREW(n)`
- `#define _GLIBCXX_ASAN_ANNOTATE_GROW(n)`
- `#define _GLIBCXX_ASAN_ANNOTATE_REINIT`
- `#define _GLIBCXX_ASAN_ANNOTATE_SHRINK(n)`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::operator!= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator<= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`constexpr bool std::operator== (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator> (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator>= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`constexpr void std::swap (vector< _Tp, _Alloc > &__x, vector< _Tp, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`std::vector (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> vector< _ValT, _Allocator >`

7.168.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

7.169 [stream_iterator.h](#) File Reference

Classes

- class [std::istream_iterator](#)< _Tp, _CharT, _Traits, _Dist >
- class [std::ostream_iterator](#)< _Tp, _CharT, _Traits >

Namespaces

- namespace [std](#)

7.169.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.170 streambuf.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _STREAMBUF_TCC`

Functions

- `template<typename _CharT, typename _Traits >`
[streamsize](#) `std::__copy_streambufs` ([basic_streambuf](#)< _CharT, _Traits > *__sbin, [basic_streambuf](#)< _CharT, _Traits > *__sbout)
- `template<typename _CharT, typename _Traits >`
[streamsize](#) `std::__copy_streambufs_eof` ([basic_streambuf](#)< _CharT, _Traits > *, [basic_streambuf](#)< _CharT, _Traits > *, bool &)

7.170.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<streambuf>`.

7.171 streambuf_iterator.h File Reference

Classes

- class [std::istreambuf_iterator](#)< _CharT, _Traits >
- class [std::ostreambuf_iterator](#)< _CharT, _Traits >

Namespaces

- namespace [std](#)

Functions

- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type` `std::__copy_move_a2` ([_CharT](#) * __first, [_CharT](#) * __last, [ostreambuf_iterator](#)< _CharT > __result)
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type` `std::__copy_move_a2` (const [_CharT](#) * __first, const [_CharT](#) * __last, [ostreambuf_iterator](#)< _CharT > __result)
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type` `std::__copy_move_a2` ([istreambuf_iterator](#)< _CharT > __first, [istreambuf_iterator](#)< _CharT > __last, [_CharT](#) * __result)
- `template<typename _CharT, typename _Size >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type` `std::__copy_n_a` ([istreambuf_iterator](#)< _CharT > __it, [_Size](#) __n, [_CharT](#) * __result, bool __strict)
- `template<typename _CharT, typename _Distance >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, void >::__type` `std::advance` ([istreambuf_iterator](#)< _CharT > &__i, [_Distance](#) __n)
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type` `std::copy` ([istreambuf_iterator](#)< _CharT > __first, [istreambuf_iterator](#)< _CharT > __last, [ostreambuf_iterator](#)< _CharT > __result)

- `template<typename _CharT >
__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT >::__type std::find
(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT &__val)`
- `template<typename _CharT, typename _Traits >
bool std::operator== (const istreambuf_iterator< _CharT, _Traits > &__a, const istreambuf_iterator< _CharT,
_Traits > &__b)`

7.171.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

7.172 string_view.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_STRING_VIEW_TCC`

7.172.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string_view>`.

7.173 string_view.tcc File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_STRING_VIEW_TCC`

7.173.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/string_view>`.

7.174 stringfwd.h File Reference

Namespaces

- namespace [std](#)

Typedefs

- typedef [basic_string](#)< char > [std::string](#)
- typedef [basic_string](#)< char16_t > [std::u16string](#)
- typedef [basic_string](#)< char32_t > [std::u32string](#)
- typedef [basic_string](#)< wchar_t > [std::wstring](#)

7.174.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

7.175 `this_thread_sleep.h` File Reference

Namespaces

- namespace `std`
- namespace `std::this_thread`

Functions

- template<typename `_Rep` , typename `_Period` >
void `std::this_thread::sleep_for` (const `chrono::duration`< `_Rep`, `_Period` > &__rtime)
- template<typename `_Clock` , typename `_Duration` >
void `std::this_thread::sleep_until` (const `chrono::time_point`< `_Clock`, `_Duration` > &__atime)

7.175.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<thread>`.

7.176 `uniform_int_dist.h` File Reference

Classes

- struct `std::uniform_int_distribution`< `_IntType` >::`param_type`
- class `std::uniform_int_distribution`< `_IntType` >

Namespaces

- namespace `std`
- namespace `std::__detail`

Concepts

- concept `std::uniform_random_bit_generator`

Functions

- template<typename `_Tp` >
constexpr bool `std::__detail::_Power_of_2` (`_Tp` __x)

7.176.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

7.177 `unique_lock.h` File Reference

Classes

- class `std::unique_lock`< `_Mutex` >

Namespaces

- namespace [std](#)

7.177.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<mutex>`.

7.178 `unique_ptr.h` File Reference

Classes

- struct [std::default_delete<_Tp>](#) >
- struct [std::default_delete<_Tp\[\]>](#)
- struct [std::hash< unique_ptr<_Tp, _Dp> >](#)
- class [std::unique_ptr<_Tp, _Dp>](#)
- class [std::unique_ptr<_Tp\[\], _Dp>](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_make_unique`

Functions

- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp>`
`bool std::operator!= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`_GLIBCXX23_CONSTEXPR bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`_GLIBCXX23_CONSTEXPR bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp>`
`_GLIBCXX23_CONSTEXPR bool std::operator< (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp>`
`_GLIBCXX23_CONSTEXPR bool std::operator<= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`_GLIBCXX23_CONSTEXPR bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`_GLIBCXX23_CONSTEXPR bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`

- template<typename _Tp, typename _Dp >
bool [std::operator==](#) (nullptr_t, const [unique_ptr](#)< _Tp, _Dp > &__x) noexcept
- template<typename _Tp, typename _Dp, typename _Up, typename _Ep >
_GLIBCXX23_CONSTEXPR bool [std::operator>](#) (const [unique_ptr](#)< _Tp, _Dp > &__x, const [unique_ptr](#)< _Up, _Ep > &__y)
- template<typename _Tp, typename _Dp >
_GLIBCXX23_CONSTEXPR bool [std::operator>](#) (const [unique_ptr](#)< _Tp, _Dp > &__x, nullptr_t)
- template<typename _Tp, typename _Dp >
_GLIBCXX23_CONSTEXPR bool [std::operator>](#) (nullptr_t, const [unique_ptr](#)< _Tp, _Dp > &__x)
- template<typename _Tp, typename _Dp, typename _Up, typename _Ep >
_GLIBCXX23_CONSTEXPR bool [std::operator>=](#) (const [unique_ptr](#)< _Tp, _Dp > &__x, const [unique_ptr](#)< _Up, _Ep > &__y)
- template<typename _Tp, typename _Dp >
_GLIBCXX23_CONSTEXPR bool [std::operator>=](#) (const [unique_ptr](#)< _Tp, _Dp > &__x, nullptr_t)
- template<typename _Tp, typename _Dp >
bool [std::operator>=](#) (nullptr_t, const [unique_ptr](#)< _Tp, _Dp > &__x)
- template<typename _Tp, typename _Dp >
[enable_if](#)<![__is_swappable](#)< _Dp >::value >::type [std::swap](#) ([unique_ptr](#)< _Tp, _Dp > &, [unique_ptr](#)< _Tp, _Dp > &)=delete

7.178.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.179 unordered_map.h File Reference

Classes

- class [std::unordered_map](#)< _Key, _Tp, _Hash, _Pred, _Alloc >
- class [std::unordered_multimap](#)< _Key, _Tp, _Hash, _Pred, _Alloc >

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_unordered_map_try_emplace`

Typedefs

- template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>, typename _Tr = __umap_traits<__cache_default<_Key, _Hash>::value>>
using [std::__umap_hashtable](#) = _Hashtable< _Key, [std::pair](#)< const _Key, _Tp >, _Alloc, __detail::__Select1st, _Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >
- template<bool _Cache>
using [std::__umap_traits](#) = __detail::__Hashtable_traits< _Cache, false, true >
- template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>, typename _Tr = __ummap_traits<__cache_default<_Key, _Hash>::value>>
using [std::__ummap_hashtable](#) = _Hashtable< _Key, [std::pair](#)< const _Key, _Tp >, _Alloc, __detail::__Select1st, _Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >

- `template<bool _Cache>`
`using std::ummap_traits = __detail::_Hashtable_traits<_Cache, false, false >`

Functions

- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool std::operator== (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool std::operator== (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void std::swap (unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void std::swap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (_InputIterator, _InputIterator, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, hash<__iter_key_t<_InputIterator>>, equal_to<__iter_key_t<_InputIterator>>, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, hash<__iter_key_t<_InputIterator>>, equal_to<__iter_key_t<_InputIterator>>, _Allocator >`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type, __Hash, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Hash, equal_to<__iter_key_t<_InputIterator>>, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash<__iter_key_t<_InputIterator>>, typename _Pred = equal_to<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_val_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type={}, __Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Hash, _Pred, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (initializer_list<pair<_Key, _Tp>>, _Allocator) -> unordered_map<_Key, _Tp, hash<_Key>, equal_to<_Key>, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (initializer_list<pair<_Key, _Tp>>, typename unordered_map<int, int>::size_type, _Allocator) -> unordered_map<_Key, _Tp, hash<_Key>, equal_to<_Key>, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (initializer_list<pair<_Key, _Tp>>, typename unordered_map<int, int>::size_type, _Hash, _Allocator) -> unordered_map<_Key, _Tp, _Hash, equal_to<_Key>, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
`std::unordered_map (initializer_list<pair<_Key, _Tp>>, typename unordered_map<int, int>::size_type={},`

```

    _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map< _Key, _Tp, _Hash, _Pred, _←
    Allocator >
• template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require←
    Allocator<_Allocator>>
    std::unordered_multimap (_InputIterator, _InputIterator, _Allocator) -> unordered_multimap< __iter_key_t<
    _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_←
    key_t< _InputIterator > >, _Allocator >
• template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require←
    Allocator<_Allocator>>
    std::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type, _Allocator)
    -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t<
    _InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >
• template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename =
    _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>
    std::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type, _Hash, ←
    _Allocator) -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Hash,
    equal_to< __iter_key_t< _InputIterator > >, _Allocator >
• template<typename _InputIterator, typename _Hash = hash< __iter_key_t< _InputIterator > >, typename _Pred = equal_to< __iter_key_←
    t< _InputIterator > >, typename _Allocator = allocator< __iter_to_alloc_t< _InputIterator > >, typename = _RequireInputIter<_InputIterator>,
    typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_←
    Allocator>>
    std::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type={}, _←
    Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multimap< __iter_key_t< _InputIterator
    >, __iter_val_t< _InputIterator >, _Hash, _Pred, _Allocator >
• template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>
    std::unordered_multimap (initializer_list< pair< _Key, _Tp > >, _Allocator) -> unordered_multimap< _Key,
    _Tp, hash< _Key >, equal_to< _Key >, _Allocator >
• template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>
    std::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type,
    _Allocator) -> unordered_multimap< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >
• template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>,
    typename = _RequireAllocator<_Allocator>>
    std::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type,
    _Hash, _Allocator) -> unordered_multimap< _Key, _Tp, _Hash, equal_to< _Key >, _Allocator >
• template<typename _Key, typename _Tp, typename _Hash = hash< _Key >, typename _Pred = equal_to< _Key >, typename _Allocator
    = allocator< pair< const _Key, _Tp > >, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>,
    typename = _RequireAllocator<_Allocator>>
    std::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type={},
    _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multimap< _Key, _Tp, _Hash, _Pred,
    _Allocator >

```

7.179.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>`.

7.180 unordered_set.h File Reference

Classes

- class `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`
- class `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`

Namespaces

- namespace [std](#)

Typedefs

- `template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::allocator<_Value>, typename _Tr = __umset_traits<__cache_default<_Value, _Hash>::value>>
using std::__umset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail::_Identity, _Pred, _Hash, __detail::_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__umset_traits = __detail::_Hashtable_traits< _Cache, true, false >`
- `template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::allocator<_Value>, typename _Tr = __uset_traits<__cache_default<_Value, _Hash>::value>>
using std::__uset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail::_Identity, _Pred, _Hash, __detail::_Mod_range_hashing, __detail::_Default_ranged_hash, __detail::_Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__uset_traits = __detail::_Hashtable_traits< _Cache, true, true >`

Functions

- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
void std::swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
void std::swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _InputIterator , typename _Allocator , typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>
std::unordered_multiset (_InputIterator, _InputIterator, unordered_multiset< int >::size_type, _Allocator) -> unordered_multiset< typename iterator_traits< _InputIterator >::value_type, hash< typename iterator_traits< _InputIterator >::value_type >, equal_to< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator , typename _Hash , typename _Allocator , typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>
std::unordered_multiset (_InputIterator, _InputIterator, unordered_multiset< int >::size_type, _Hash, _Allocator) -> unordered_multiset< typename iterator_traits< _InputIterator >::value_type, _Hash, equal_to< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator , typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>
std::unordered_multiset (_InputIterator, _InputIterator, unordered_multiset< int >::size_type={}, _Hash=Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multiset< typename iterator_traits< _InputIterator >::value_type, _Hash, _Pred, _Allocator >`
- `template<typename _Tp , typename _Allocator , typename = _RequireAllocator<_Allocator>>
std::unordered_multiset (initializer_list< _Tp >, unordered_multiset< int >::size_type, _Allocator) -> unordered_multiset< _Tp, hash< _Tp >, equal_to< _Tp >, _Allocator >`

- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
`std::unordered_multiset (initializer_list< _Tp >, unordered_multiset< int >::size_type, _Hash, _Allocator) ->`
`unordered_multiset< _Tp, _Hash, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>,`
`typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_`
`Allocator>>`
`std::unordered_multiset (initializer_list< _Tp >, unordered_multiset< int >::size_type={}, _Hash=_Hash(), _`
`_Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multiset< _Tp, _Hash, _Pred, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require`
`Allocator<_Allocator>>`
`std::unordered_set (_InputIterator, _InputIterator, unordered_set< int >::size_type, _Allocator) ->`
`unordered_set< typename iterator_traits< _InputIterator >::value_type, hash< typename iterator_traits< _`
`_InputIterator >::value_type >, equal_to< typename iterator_traits< _InputIterator >::value_type >, _Allocator`
`>`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
`std::unordered_set (_InputIterator, _InputIterator, unordered_set< int >::size_type, _Hash, _Allocator) -`
`> unordered_set< typename iterator_traits< _InputIterator >::value_type, _Hash, equal_to< typename`
`iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred`
`= equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_Input`
`Iterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename`
`= _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
`std::unordered_set (_InputIterator, _InputIterator, unordered_set< int >::size_type={}, _Hash=_Hash(), _`
`Pred=_Pred(), _Allocator=_Allocator()) -> unordered_set< typename iterator_traits< _InputIterator >::value`
`_type, _Hash, _Pred, _Allocator >`
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::unordered_set (initializer_list< _Tp >, unordered_set< int >::size_type, _Allocator) -> unordered_set<`
`_Tp, hash< _Tp >, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename =`
`_RequireAllocator<_Allocator>>`
`std::unordered_set (initializer_list< _Tp >, unordered_set< int >::size_type, _Hash, _Allocator) ->`
`unordered_set< _Tp, _Hash, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>,`
`typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_`
`Allocator>>`
`std::unordered_set (initializer_list< _Tp >, unordered_set< int >::size_type={}, _Hash=_Hash(), _Pred=_`
`Pred(), _Allocator=_Allocator()) -> unordered_set< _Tp, _Hash, _Pred, _Allocator >`

7.180.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_set>`.

7.181 uses_allocator_args.h File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_make_obj_using_allocator`

Functions

- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp std::make_obj_using_allocator (const _Alloc &__a, _Args &&... __args)`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`constexpr _Tp * std::uninitialized_construct_using_allocator (_Tp *__p, const _Alloc &__a, _Args &&... __args)`
- `template<_Std_pair _Tp, typename _Alloc >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, _Up &&, _Vp &&) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, const pair< _Up, _Vp > &) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Up, typename _Vp >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &, pair< _Up, _Vp > &&) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`requires (! _Std_pair<_Tp>)`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &__a, _Args &&... __args) noexcept`
- `template<_Std_pair _Tp, typename _Alloc, typename _Tuple1, typename _Tuple2 >`
`constexpr auto std::uses_allocator_construction_args (const _Alloc &__a, piecewise_construct_t, _Tuple1 &&__x, _Tuple2 &&__y) noexcept`

7.181.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.182 utility.h File Reference

Classes

- struct [std::integer_sequence](#)< _Tp, _Idx >

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_integer_sequence`
- `#define __cpp_lib_tuple_element_t`

Typedefs

- `template<typename _Tp, typename _Up = typename remove_cv<_Tp>::type, typename = typename enable_if<is_same<_Tp, _Up><::value>::type, size_t = tuple_size<_Tp>::value>`
`using std::_enable_if_has_tuple_size = _Tp`
- `template<typename _Tp >`
`using std::_is_in_place_type = bool_constant< __is_in_place_type_v< _Tp > >`
- `template<size_t __i, typename _Tp >`
`using std::_tuple_element_t = typename tuple_element< __i, _Tp >::type`
- `template<size_t... _Idx>`
`using std::index_sequence = integer_sequence< size_t, _Idx... >`
- `template<typename... _Types>`
`using std::index_sequence_for = make_index_sequence< sizeof...(_Types)>`

- `template<size_t _Num>`
`using std::make_index_sequence = make_integer_sequence< size_t, _Num >`
- `template<typename _Tp, _Tp _Num>`
`using std::make_integer_sequence = __make_integer_seq< integer_sequence, _Tp, _Num >`
- `template<size_t __i, typename _Tp >`
`using std::tuple_element_t = typename tuple_element< __i, _Tp >::type`

Functions

- `template<typename _Tp, typename... _Types>`
`constexpr size_t std::__find_uniq_type_in_pack ()`

Variables

- `template<typename >`
`constexpr bool std::__is_in_place_type_v`
- `template<typename _Tp >`
`constexpr bool std::__is_in_place_type_v< in_place_type_t< _Tp > >`
- `constexpr in_place_t std::in_place`
- `template<size_t _Idx>`
`constexpr in_place_index_t< _Idx > std::in_place_index`
- `template<typename _Tp >`
`constexpr in_place_type_t< _Tp > std::in_place_type`
- `template<typename _Tp >`
`constexpr size_t std::tuple_size_v`

7.182.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

This file contains the parts of `<utility>` needed by other headers, so they don't need to include the whole of `<utility>`.

7.183 `valarray_after.h` File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define _DEFINE_EXPR_BINARY_FUNCTION(_Fun, _UFun)`
- `#define _DEFINE_EXPR_BINARY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_EXPR_UNARY_FUNCTION(_Name, _UName)`
- `#define _DEFINE_EXPR_UNARY_OPERATOR(_Op, _Name)`

Functions

- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Abs, _Expr, _Dom >, typename _Dom::value_type > std::abs (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Abs, _ValArray, _Tp >, _Tp > std::abs (const valarray< _Tp > &__v)`

- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Acos, _Expr, _Dom >, typename _Dom::value_type > std::acos (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Acos, _ValArray, _Tp >, _Tp > std::acos (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Asin, _Expr, _Dom >, typename _Dom::value_type > std::asin (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Asin, _ValArray, _Tp >, _Tp > std::asin (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Atan, _Expr, _Dom >, typename _Dom::value_type > std::atan (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Atan, _ValArray, _Tp >, _Tp > std::atan (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Atan2, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename ↵`
`_Dom::value_type > std::atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename`
`_Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Atan2, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename ↵`
`_Dom::value_type > std::atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray<`
`typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::_Atan2, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > std↵`
`::atan2 (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename ↵`
`_Dom2::value_type > &__e2)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Atan2, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename ↵`
`_Dom::value_type > std::atan2 (const typename _Dom::value_type &__t, const _Expr< _Dom, typename ↵`
`_Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Atan2, _Constant, _ValArray, _Tp, _Tp >, _Tp > std::atan2 (const typename`
`valarray< _Tp >::value_type &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Atan2, _ValArray, _Constant, _Tp, _Tp >, _Tp > std::atan2 (const valarray<`
`_Tp > &__v, const typename valarray< _Tp >::value_type &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::_Atan2, _ValArray, _ValArray, _Tp, _Tp >, _Tp > std::atan2 (const valarray< _Tp`
`> &__v, const valarray< _Tp > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Atan2, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename ↵`
`_Dom::value_type > std::atan2 (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Cos, _Expr, _Dom >, typename _Dom::value_type > std::cos (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Cos, _ValArray, _Tp >, _Tp > std::cos (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Cosh, _Expr, _Dom >, typename _Dom::value_type > std::cosh (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`

- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Cosh, _ValArray, _Tp >, _Tp > std::cosh (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Exp, _Expr, _Dom >, typename _Dom::value_type > std::exp (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Exp, _ValArray, _Tp >, _Tp > std::exp (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Log, _Expr, _Dom >, typename _Dom::value_type > std::log (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Log, _ValArray, _Tp >, _Tp > std::log (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Log10, _Expr, _Dom >, typename _Dom::value_type > std::log10 (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Log10, _ValArray, _Tp >, _Tp > std::log10 (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_NotEqualTo, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::_NotEqualTo, typename _Dom::value_type >::result_type > std::operator!= (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_NotEqualTo, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::_NotEqualTo, typename _Dom::value_type >::result_type > std::operator!= (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::_NotEqualTo, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::_NotEqualTo, typename _Dom1::value_type >::result_type > std::operator!= (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_NotEqualTo, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_NotEqualTo, typename _Dom::value_type >::result_type > std::operator!= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_NotEqualTo, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_NotEqualTo, typename _Dom::value_type >::result_type > std::operator!= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Modulus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::_Modulus, typename _Dom::value_type >::result_type > std::operator% (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Modulus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::_Modulus, typename _Dom::value_type >::result_type > std::operator% (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::_Modulus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::_Modulus, typename _Dom1::value_type >::result_type > std::operator% (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::_Modulus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::_Modulus, typename _Dom::value_type >::result_type > std::operator% (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__modulus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__modulus, typename _Dom::value_type >::result_type > std::operator% (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const`
`_Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const`
`_Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__bitwise_and, typename _Dom1::value_type >::result_type > std::operator& (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const`
`typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const`
`valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__logical_and, typename _Dom::value_type >::result_type > std::operator&& (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__logical_and, typename _Dom::value_type >::result_type > std::operator&& (const`
`_Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__logical_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__logical_and, typename _Dom1::value_type >::result_type > std::operator&& (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__logical_and, typename _Dom::value_type >::result_type > std::operator&& (const`
`typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__logical_and, typename _Dom::value_type >::result_type > std::operator&& (const`
`valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__multiplies, typename _Dom::value_type >::result_type > std::operator* (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__multiplies, typename _Dom::value_type >::result_type > std::operator* (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__multiplies, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__multiplies, typename _Dom1::value_type >::result_type > std::operator* (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__multiplies, typename _Dom::value_type >::result_type > std::operator* (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__multiplies, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__multiplies, typename _Dom::value_type >::result_type > std::operator* (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__plus, typename _Dom::value_type >::result_type > std::operator+ (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__plus, typename _Dom::value_type >::result_type > std::operator+ (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__plus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__plus, typename _Dom1::value_type >::result_type > std::operator+ (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__plus, typename _Dom::value_type >::result_type > std::operator+ (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__plus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__plus, typename _Dom::value_type >::result_type > std::operator+ (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< struct std::__minus, typename _Dom::value_type >::result_type > std::operator- (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< struct std::__minus, typename _Dom::value_type >::result_type > std::operator- (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__minus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__minus, typename _Dom1::value_type >::result_type > std::operator- (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__minus, typename _Dom::value_type >::result_type > std::operator- (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__minus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< struct std::__minus, typename _Dom::value_type >::result_type > std::operator- (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator/ (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator/ (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__divides, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type >::result_type > std::operator/ (const _Expr< _Dom1,`
`typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > std::operator/ (const typename _Dom,`
`typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__divides, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > std::operator/ (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator< (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator< (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__less, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type >::result_type > std::operator< (const _Expr< _Dom1,`
`typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > std::operator< (const typename _Dom,`
`typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type >::result_type > std::operator< (const valarray< type-`
`name _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_left, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator<< (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_left, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type >::result_type > std::operator<< (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__shift_left, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type >::result_type > std::operator<< (const _Expr< _Dom1,`
`typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_left, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__shift_left, typename _Dom::value_type >::result_type > std::operator<< (const typename`
`_Dom::value_type & __t, const _Expr< _Dom, typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_left, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__shift_left, typename _Dom::value_type >::result_type > std::operator<< (const valarray<`
`typename _Dom::value_type > & __v, const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__less_equal, typename _Dom::value_type >::result_type > std::operator<= (const _Expr<`
`_Dom, typename _Dom::value_type > & __e, const valarray< typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__less_equal, typename _Dom::value_type >::result_type > std::operator<= (const _Expr<`
`_Dom, typename _Dom::value_type > & __v, const typename _Dom::value_type & __t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__less_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename _fun< struct std::__`
`less_equal, typename _Dom1::value_type >::result_type > std::operator<= (const _Expr< _Dom1, typename`
`_Dom1::value_type > & __v, const _Expr< _Dom2, typename _Dom2::value_type > & __w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name _fun< struct std::__less_equal, typename _Dom::value_type >::result_type > std::operator<= (const`
`typename _Dom::value_type & __t, const _Expr< _Dom, typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__less_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name _fun< struct std::__less_equal, typename _Dom::value_type >::result_type > std::operator<= (const`
`valarray< typename _Dom::value_type > & __v, const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__equal_to, typename _Dom::value_type >::result_type > std::operator== (const _Expr<`
`_Dom, typename _Dom::value_type > & __e, const valarray< typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__equal_to, typename _Dom::value_type >::result_type > std::operator== (const _Expr<`
`_Dom, typename _Dom::value_type > & __v, const typename _Dom::value_type & __t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename _fun< struct std::__`
`equal_to, typename _Dom1::value_type >::result_type > std::operator== (const _Expr< _Dom1, typename`
`_Dom1::value_type > & __v, const _Expr< _Dom2, typename _Dom2::value_type > & __w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__equal_to, typename _Dom::value_type >::result_type > std::operator== (const typename`
`_Dom::value_type & __t, const _Expr< _Dom, typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`_fun< struct std::__equal_to, typename _Dom::value_type >::result_type > std::operator== (const valarray<`
`typename _Dom::value_type > & __v, const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`_fun< struct std::__greater, typename _Dom::value_type >::result_type > std::operator> (const _Expr< _Dom,`
`typename _Dom::value_type > & __e, const valarray< typename _Dom::value_type > & __v)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__greater, typename _Dom::value_type >::result_type > std::operator> (const _Expr< _↵`
`Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__greater, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__↵`
`greater, typename _Dom1::value_type >::result_type > std::operator> (const _Expr< _Dom1, typename _↵`
`Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__greater, typename _Dom::value_type >::result_type > std::operator> (const typename`
`_Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename ↵`
`__fun< struct std::__greater, typename _Dom::value_type >::result_type > std::operator> (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > std::operator>=`
`(const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type >`
`&__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, ↵`
`typename __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > std::operator>=`
`(const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__greater_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__↵`
`greater_equal, typename _Dom1::value_type >::result_type > std::operator>= (const _Expr< _Dom1, type-`
`name _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, ↵`
`typename __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > std::operator>=`
`(const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__greater_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__greater_equal, typename _Dom::value_type >::result_type > std::operator>=`
`(const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type >`
`&__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__shift_right, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__↵`
`shift_right, typename _Dom1::value_type >::result_type > std::operator>> (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const type-`
`name _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< struct std::__shift_right, _ValArray, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const`
`valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _Constant, _Dom, typename _Dom::value_type >, type-`
`name __fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const`
`_Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__bitwise_xor, typename _Dom1::value_type >::result_type > std::operator^ (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _Constant, _Expr, typename _Dom::value_type, _Dom >, type-`
`name __fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const`
`typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_xor, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__bitwise_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__bitwise_or, typename _Dom1::value_type >::result_type > std::operator| (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const typename`
`_Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__bitwise_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const _Expr<`
`_Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< struct std::__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const _Expr<`
`_Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__logical_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< struct std::__`
`__logical_or, typename _Dom1::value_type >::result_type > std::operator|| (const _Expr< _Dom1, typename`
`_Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const typename`
`_Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__logical_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< struct std::__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const valarray<`
`typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`_Dom::value_type > std::pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename`
`_Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom`
`::value_type > std::pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename`
`_Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< struct std::__Pow, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > std`
`::pow (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename`
`_Dom2::value_type > &__e2)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`_Dom::value_type > std::pow (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom`
`::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::__Pow, _Constant, _ValArray, _Tp, _Tp >, _Tp > std::pow (const typename`
`valarray< _Tp >::value_type &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::__Pow, _ValArray, _Constant, _Tp, _Tp >, _Tp > std::pow (const valarray< _Tp`
`> &__v, const typename valarray< _Tp >::value_type &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< struct std::__Pow, _ValArray, _ValArray, _Tp, _Tp >, _Tp > std::pow (const valarray< _Tp`
`> &__v, const valarray< _Tp > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< struct std::__Pow, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom`
`::value_type > std::pow (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename`
`_Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::__Sin, _Expr, _Dom >, typename _Dom::value_type > std::sin (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::__Sin, _ValArray, _Tp >, _Tp > std::sin (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::__Sinh, _Expr, _Dom >, typename _Dom::value_type > std::sinh (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::__Sinh, _ValArray, _Tp >, _Tp > std::sinh (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::__Sqrt, _Expr, _Dom >, typename _Dom::value_type > std::sqrt (const _Expr<`
`_Dom, typename _Dom::value_type > &__e)`

- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Sqrt, _ValArray, _Tp >, _Tp > std::sqrt (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Tan, _Expr, _Dom >, typename _Dom::value_type > std::tan (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Tan, _ValArray, _Tp >, _Tp > std::tan (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< struct std::_Tanh, _Expr, _Dom >, typename _Dom::value_type > std::tanh (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< struct std::_Tanh, _ValArray, _Tp >, _Tp > std::tanh (const valarray< _Tp > &__v)`

7.183.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.184 `valarray_array.h` File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _DEFINE_ARRAY_FUNCTION(_Op, _Name)`

Functions

- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s1, _Array< _Tp > __b, size_t __s2)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __src, size_t __n, _Array< size_t > __i, _Array< _Tp > __dst, _Array< size_t > __j)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, _Tp *__restrict __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b, const size_t *__restrict __i)`

- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __src, size_t __n, const size_t *__restrict __i, _Tp *__restrict __dst, const size_t *__restrict __j)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __src, size_t __n, size_t __s1, _Tp *__restrict __dst, size_t __s2)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__b, const _Tp *__e, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __o, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void std::__valarray_default_construct (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void std::__valarray_destroy_elements (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, _Array< size_t > __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, const size_t *__restrict __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill_construct (_Tp *__b, _Tp *__e, const _Tp &__t)`
- `template<typename _Tp >`
`_Tp * std::__valarray_get_storage (size_t)`
- `template<typename _Ta >`
`_Ta::value_type std::__valarray_max (const _Ta &__a)`
- `template<typename _Ta >`
`_Ta::value_type std::__valarray_min (const _Ta &__a)`
- `void std::__valarray_release_memory (void *__p)`
- `template<typename _Tp >`
`_Tp std::__valarray_sum (const _Tp *__f, const _Tp *__l)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`

- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`

- `template<typename _Tp >`
`void std::Array_augmented_bitwise_or (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_or (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_bitwise_or (_Array< _Tp > __a, size_t __s, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __s, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented_divides (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented_divides (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_divides (_Array< _Tp > __a, _Array< bool > __m, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented_divides (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented_divides (_Array< _Tp > __a, _Array< size_t > __i, const Expr< _Dom, _Tp > &__e, size_t __n)`

- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__divides (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__divides (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`

- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool`
`> __m)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t >`
`__i)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool`
`> __m)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, const _Tp &__t)`

- `template<typename _Tp >`
`void std::__Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__multiplies (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__plus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented__shift_left (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`

- `template<typename _Tp >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__shift_left (_Array< _Tp > __a, size_t __s, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__shift_right (_Array< _Tp > __a, size_t __s, const Expr< _Dom, _Tp > &__e, size_t __n)`

7.184.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.185 valarray_array.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _VALARRAY_ARRAY_TCC`

Functions

- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< bool > __m, size_t __n, _Array< _Tp > __b, _Array< bool > __k)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __e, _Array< size_t > __f, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, size_t __s)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy_construct (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, _Array< bool > __m, const _Tp &__t)`

7.185.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.186 valarray_before.h File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

7.186.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

7.187 vector.tcc File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _VECTOR_TCC`

7.187.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

7.188 bitset File Reference

Classes

- struct [std::_Base_bitset<_Nw>](#)
- struct [std::_Base_bitset<0>](#)
- struct [std::_Base_bitset<1>](#)
- class [std::bitset<_Nb>](#)
- struct [std::hash<::bitset<_Nb>>](#)
- class [std::bitset<_Nb>::reference](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_BITSET`
- `#define _GLIBCXX_BITSET_BITS_PER_ULL`
- `#define _GLIBCXX_BITSET_BITS_PER_WORD`
- `#define _GLIBCXX_BITSET_WORDS(__n)`

Functions

- `template<size_t _Nb>`
[bitset<_Nb>](#) [std::operator&](#) (const [bitset<_Nb>](#) &__x, const [bitset<_Nb>](#) &__y) noexcept
- `template<size_t _Nb>`
[bitset<_Nb>](#) [std::operator^](#) (const [bitset<_Nb>](#) &__x, const [bitset<_Nb>](#) &__y) noexcept
- `template<size_t _Nb>`
[bitset<_Nb>](#) [std::operator|](#) (const [bitset<_Nb>](#) &__x, const [bitset<_Nb>](#) &__y) noexcept
- `template<class _CharT, class _Traits, size_t _Nb>`
[std::basic_ostream<_CharT, _Traits>](#) & [std::operator<<](#) ([std::basic_ostream<_CharT, _Traits>](#) &__os, const [bitset<_Nb>](#) &__x)
- `template<class _CharT, class _Traits, size_t _Nb>`
[std::basic_istream<_CharT, _Traits>](#) & [std::operator>>](#) ([std::basic_istream<_CharT, _Traits>](#) &__is, [bitset<_Nb>](#) &__x)

7.188.1 Detailed Description

This is a Standard C++ Library header.

7.189 `bitset` File Reference

Classes

- class `std::__debug::bitset<_Nb>`
- struct `std::hash<__debug::bitset<_Nb>>`

Namespaces

- namespace `std`
- namespace `std::__debug`

Functions

- `template<size_t_Nb>`
`bitset<_Nb> std::__debug::operator& (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<typename _CharT, typename _Traits, size_t_Nb>`
`std::basic_ostream<_CharT, _Traits> & std::__debug::operator<< (std::basic_ostream<_CharT, _Traits> &__os, const bitset<_Nb> &__x)`
- `template<typename _CharT, typename _Traits, size_t_Nb>`
`std::basic_istream<_CharT, _Traits> & std::__debug::operator>> (std::basic_istream<_CharT, _Traits> &__is, bitset<_Nb> &__x)`
- `template<size_t_Nb>`
`bitset<_Nb> std::__debug::operator^ (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<size_t_Nb>`
`bitset<_Nb> std::__debug::operator| (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`

7.189.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.190 `cassert` File Reference

7.190.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `assert.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.191 `complex` File Reference

Macros

- `#define _GLIBCXX_CCOMPLEX`

7.191.1 Detailed Description

This is a Standard C++ Library header.

7.192 ccomplex File Reference

Macros

- `#define _GLIBCXX_TR1_CCOMPLEX`

7.192.1 Detailed Description

This is a TR1 C++ Library header.

7.193 cctype File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CCTYPE`

7.193.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `cctype.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.194 cctype File Reference

Macros

- `#define _GLIBCXX_TR1_CCTYPE`

7.194.1 Detailed Description

This is a TR1 C++ Library header.

7.195 cerrno File Reference

Macros

- `#define _GLIBCXX_CERRNO`
- `#define errno`

7.195.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `errno.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.196 cenv File Reference

Macros

- `#define _GLIBCXX_CFENV`

7.196.1 Detailed Description

This is a Standard C++ Library header.

7.197 cenv File Reference

Macros

- `#define _GLIBCXX_TR1_CENV`

7.197.1 Detailed Description

This is a TR1 C++ Library header.

7.198 cfloat File Reference

Macros

- `#define _GLIBCXX_CFLOAT`
- `#define DECIMAL_DIG`
- `#define FLT_EVAL_METHOD`

7.198.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `float.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.199 cfloat File Reference

Macros

- `#define _GLIBCXX_TR1_CFLOAT`

7.199.1 Detailed Description

This is a TR1 C++ Library header.

7.200 charconv File Reference

Classes

- struct [std::from_chars_result](#)
- struct [std::to_chars_result](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define _GLIBCXX_CHARCONV`
- `#define _GLIBCXX_TO_CHARS(T)`

Typedefs

- `template<typename _Tp >`
`using std::__detail::__integer_from_chars_result_type = enable_if_t<__or_<__is_signed_integer<_Tp>, __is_unsigned_integer<_Tp>, is_same<char, remove_cv_t<_Tp>>>::value, from_chars_result>`
- `template<typename _Tp >`
`using std::__detail::__integer_to_chars_result_type = enable_if_t<__or_<__is_signed_integer<_Tp>, __is_unsigned_integer<_Tp>, is_same<char, remove_cv_t<_Tp>>>::value, to_chars_result>`
- `template<typename _Tp >`
`using std::__detail::__unsigned_least_t = typename __to_chars_unsigned_type<_Tp>::type`

Enumerations

- enum class `std::chars_format` { `scientific`, `fixed`, `hex`, `general` }

Functions

- `template<bool _DecOnly, typename _Tp >`
`bool std::__detail::__from_chars_alnum (const char *__first, const char *__last, _Tp &__val, int __base)`
- `template<bool _DecOnly = false>`
`unsigned char std::__detail::__from_chars_alnum_to_val (unsigned char __c)`
- `constexpr auto std::__detail::__from_chars_alnum_to_val_table ()`
- `template<bool _DecOnly, typename _Tp >`
`bool std::__detail::__from_chars_pow2_base (const char *__first, const char *__last, _Tp &__val, int __base)`
- `template<typename _Tp >`
`bool std::__detail::__raise_and_add (_Tp &__val, int __base, unsigned char __c)`
- `template<typename _Tp >`
`to_chars_result std::__detail::__to_chars (char *__first, char *__last, _Tp __val, int __base) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type<_Tp> std::__detail::__to_chars_10 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type<_Tp> std::__detail::__to_chars_16 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type<_Tp> std::__detail::__to_chars_2 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__integer_to_chars_result_type<_Tp> std::__detail::__to_chars_8 (char *__first, char *__last, _Tp __val) noexcept`
- `template<typename _Tp >`
`__detail::__integer_to_chars_result_type<_Tp> std::__to_chars_i (char *__first, char *__last, _Tp __value, int __base=10)`
- `template<typename _Tp >`
`constexpr unsigned std::__detail::__to_chars_len (_Tp __value, int __base) noexcept`
- `template<typename _Tp >`
`constexpr unsigned std::__detail::__to_chars_len_2 (_Tp __value) noexcept`
- `template<typename _Tp >`
`__detail::__integer_from_chars_result_type<_Tp> std::from_chars (const char *__first, const char *__last, _Tp &__value, int __base=10)`
- `constexpr chars_format std::operator& (chars_format __lhs, chars_format __rhs) noexcept`
- `constexpr chars_format & std::operator&= (chars_format &__lhs, chars_format __rhs) noexcept`
- `constexpr chars_format std::operator^ (chars_format __lhs, chars_format __rhs) noexcept`
- `constexpr chars_format & std::operator^= (chars_format &__lhs, chars_format __rhs) noexcept`

- constexpr [chars_format](#) **std::operator|** ([chars_format](#) __lhs, [chars_format](#) __rhs) noexcept
- constexpr [chars_format](#) & **std::operator|=** ([chars_format](#) & __lhs, [chars_format](#) __rhs) noexcept
- constexpr [chars_format](#) **std::operator~** ([chars_format](#) __fmt) noexcept
- [to_chars_result](#) **std::to_chars** (char *, char *, bool, int=10)=delete
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, char __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, signed char __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, signed int __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, signed long __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, signed long long __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, signed short __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, unsigned char __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, unsigned int __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, unsigned long __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, unsigned long long __value, int __base=10)
- [to_chars_result](#) **std::to_chars** (char * __first, char * __last, unsigned short __value, int __base=10)

7.200.1 Detailed Description

This is a Standard C++ Library header.

7.201 chrono File Reference

Namespaces

- namespace [std](#)
- namespace [std::chrono](#)
- namespace [std::literals](#)
- namespace [std::literals::chrono_literals](#)

Macros

- #define [_GLIBCXX_CHRONO](#)

Typedefs

- using [std::chrono::__detail::__months_years_conversion_disambiguator](#) = void
- using [std::chrono::gps_seconds](#) = [gps_time](#)< [seconds](#) >
- template<typename [_Duration](#) >
using [std::chrono::gps_time](#) = [time_point](#)< [gps_clock](#), [_Duration](#) >
- using [std::chrono::local_days](#) = [local_time](#)< [days](#) >
- using [std::chrono::local_seconds](#) = [local_time](#)< [seconds](#) >
- template<typename [_Duration](#) >
using [std::chrono::local_time](#) = [time_point](#)< [local_t](#), [_Duration](#) >
- using [std::chrono::tai_seconds](#) = [tai_time](#)< [seconds](#) >
- template<typename [_Duration](#) >
using [std::chrono::tai_time](#) = [time_point](#)< [tai_clock](#), [_Duration](#) >
- using [std::chrono::utc_seconds](#) = [utc_time](#)< [seconds](#) >
- template<typename [_Duration](#) >
using [std::chrono::utc_time](#) = [time_point](#)< [utc_clock](#), [_Duration](#) >

Functions

- constexpr unsigned **std::chrono::__detail::__modulo** (long long __n, unsigned __d)
- constexpr long long **std::chrono::__detail::__pow10** (unsigned __n)
- constexpr bool **std::chrono::is_am** (const [hours](#) &__h) noexcept
- constexpr bool **std::chrono::is_pm** (const [hours](#) &__h) noexcept
- constexpr [hours](#) **std::chrono::make12** (const [hours](#) &__h) noexcept
- constexpr [hours](#) **std::chrono::make24** (const [hours](#) &__h, bool __is_pm) noexcept
- constexpr chrono::day **std::literals::chrono_literals::operator""d** (unsigned long long __d) noexcept
- constexpr chrono::year **std::literals::chrono_literals::operator""y** (unsigned long long __y) noexcept
- template<typename _CharT, typename _Traits, typename _Rep, typename _Period >
[basic_ostream](#)< _CharT, _Traits > & **std::chrono::operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os,
const [duration](#)< _Rep, _Period > &__d)

Variables

- constexpr unsigned **std::chrono::__detail::__days_per_month** [12]
- constexpr month **std::chrono::April**
- constexpr month **std::chrono::August**
- constexpr month **std::chrono::December**
- constexpr month **std::chrono::February**
- constexpr weekday **std::chrono::Friday**
- template<> constexpr bool **std::chrono::is_clock_v**< [gps_clock](#) >
- template<> constexpr bool **std::chrono::is_clock_v**< [tai_clock](#) >
- template<> constexpr bool **std::chrono::is_clock_v**< [utc_clock](#) >
- constexpr month **std::chrono::January**
- constexpr month **std::chrono::July**
- constexpr month **std::chrono::June**
- constexpr last_spec **std::chrono::last**
- constexpr month **std::chrono::March**
- constexpr month **std::chrono::May**
- constexpr weekday **std::chrono::Monday**
- constexpr month **std::chrono::November**
- constexpr month **std::chrono::October**
- constexpr weekday **std::chrono::Saturday**
- constexpr month **std::chrono::September**
- constexpr weekday **std::chrono::Sunday**
- constexpr weekday **std::chrono::Thursday**
- constexpr weekday **std::chrono::Tuesday**
- constexpr weekday **std::chrono::Wednesday**

7.201.1 Detailed Description

This is a Standard C++ Library header.

7.202 chrono File Reference

Namespaces

- namespace [std](#)
- namespace [std::chrono](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_CHRONO`

Variables

- `template<typename _Rep >`
`constexpr bool std::chrono::experimental::treat_as_floating_point_v`

7.202.1 Detailed Description

This is a TS C++ Library header.

7.203 cinttypes File Reference**Macros**

- `#define _GLIBCXX_CINTTYPES`

7.203.1 Detailed Description

This is a Standard C++ Library header.

7.204 cinttypes File Reference**Macros**

- `#define _GLIBCXX_TR1_CINTTYPES`

7.204.1 Detailed Description

This is a TR1 C++ Library header.

7.205 ciso646 File Reference**7.205.1 Detailed Description**

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `iso646.h`, which is empty in C++.

7.206 climits File Reference**Macros**

- `#define _GLIBCXX_CLIMITS`
- `#define LLONG_MAX`
- `#define LLONG_MIN`
- `#define ULLONG_MAX`

7.206.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `limits.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.207 climits File Reference

Macros

- `#define _GLIBCXX_TR1_CLIMITS`

7.207.1 Detailed Description

This is a TR1 C++ Library header.

7.208 clocale File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CLOCALE`

7.208.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `locale.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.209 cmath File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_hypot`
- `#define __cpp_lib_interpolate`
- `#define _GLIBCXX_CMATH`
- `#define _GLIBCXX_INCLUDE_NEXT_C_HEADERS`

Functions

- `template<typename _Tp >`
`_Tp std::__hypot3 (_Tp __x, _Tp __y, _Tp __z)`
- `template<typename _Fp >`
`constexpr _Fp std::__lerp (_Fp __a, _Fp __b, _Fp __t) noexcept`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::acos (_Tp __x)`
- `constexpr float std::acos (float __x)`
- `constexpr long double std::acos (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::asin (_Tp __x)`
- `constexpr float std::asin (float __x)`
- `constexpr long double std::asin (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::atan (_Tp __x)`

- constexpr float **std::atan** (float __x)
- constexpr long double **std::atan** (long double __x)
- template<typename _Tp, typename _Up >
constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type **std::atan2** (_Tp __y, _Up __x)
- constexpr float **std::atan2** (float __y, float __x)
- constexpr long double **std::atan2** (long double __y, long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::ceil** (_Tp __x)
- constexpr float **std::ceil** (float __x)
- constexpr long double **std::ceil** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::cos** (_Tp __x)
- constexpr float **std::cos** (float __x)
- constexpr long double **std::cos** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::cosh** (_Tp __x)
- constexpr float **std::cosh** (float __x)
- constexpr long double **std::cosh** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::exp** (_Tp __x)
- constexpr float **std::exp** (float __x)
- constexpr long double **std::exp** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::fabs** (_Tp __x)
- constexpr float **std::fabs** (float __x)
- constexpr long double **std::fabs** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::floor** (_Tp __x)
- constexpr float **std::floor** (float __x)
- constexpr long double **std::floor** (long double __x)
- template<typename _Tp, typename _Up >
constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type **std::fmod** (_Tp __x, _Up __y)
- constexpr float **std::fmod** (float __x, float __y)
- constexpr long double **std::fmod** (long double __x, long double __y)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::frexp** (_Tp __x, int * __exp)
- float **std::frexp** (float __x, int * __exp)
- long double **std::frexp** (long double __x, int * __exp)
- template<typename _Tp, typename _Up, typename _Vp >
__gnu_cxx::__promoted_t< _Tp, _Up, _Vp > **std::hypot** (_Tp __x, _Up __y, _Vp __z)
- double **std::hypot** (double __x, double __y, double __z)
- float **std::hypot** (float __x, float __y, float __z)
- long double **std::hypot** (long double __x, long double __y, long double __z)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::ldexp** (_Tp __x, int __exp)
- constexpr float **std::ldexp** (float __x, int __exp)
- constexpr long double **std::ldexp** (long double __x, int __exp)
- template<typename _Tp, typename _Up, typename _Vp >
constexpr __gnu_cxx::__promoted_t< _Tp, _Up, _Vp > **std::lerp** (_Tp __x, _Up __y, _Vp __z) noexcept
- constexpr double **std::lerp** (double __a, double __b, double __t) noexcept

- constexpr float **std::lerp** (float __a, float __b, float __t) noexcept
- constexpr long double **std::lerp** (long double __a, long double __b, long double __t) noexcept
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::log** (_Tp __x)
- constexpr float **std::log** (float __x)
- constexpr long double **std::log** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::log10** (_Tp __x)
- constexpr float **std::log10** (float __x)
- constexpr long double **std::log10** (long double __x)
- float **std::modf** (float __x, float *__iptr)
- long double **std::modf** (long double __x, long double *__iptr)
- template<typename _Tp, typename _Up >
constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type **std::pow** (_Tp __x, _Up __y)
- constexpr float **std::pow** (float __x, float __y)
- constexpr long double **std::pow** (long double __x, long double __y)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::sin** (_Tp __x)
- constexpr float **std::sin** (float __x)
- constexpr long double **std::sin** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::sinh** (_Tp __x)
- constexpr float **std::sinh** (float __x)
- constexpr long double **std::sinh** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::sqrt** (_Tp __x)
- constexpr float **std::sqrt** (float __x)
- constexpr long double **std::sqrt** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::tan** (_Tp __x)
- constexpr float **std::tan** (float __x)
- constexpr long double **std::tan** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::tanh** (_Tp __x)
- constexpr float **std::tanh** (float __x)
- constexpr long double **std::tanh** (long double __x)

7.209.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the *.h implementation files.

This is the C++ version of the Standard C Library header `math.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.210 cmath File Reference

Namespaces

- namespace [`__gnu_cxx`](#)

Macros

- `#define` `_EXT_CMATH`

7.210.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.211 cmath File Reference

Namespaces

- namespace [std](#)
- namespace [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_CMATH`

Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_laguerre` (unsigned int __n, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_laguerref` (unsigned int __n, unsigned int __m, float __x)
- `long double std::tr1::assoc_laguerrel` (unsigned int __n, unsigned int __m, long double __x)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_legendre` (unsigned int __l, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_legendref` (unsigned int __l, unsigned int __m, float __x)
- `long double std::tr1::assoc_legendrel` (unsigned int __l, unsigned int __m, long double __x)
- `template<typename _Tpx, typename _Tpy >`
`__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type std::tr1::beta` (_Tpx __x, _Tpy __y)
- `float std::tr1::betaf` (float __x, float __y)
- `long double std::tr1::betal` (long double __x, long double __y)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_1` (_Tp __k)
- `float std::tr1::comp_ellint_1f` (float __k)
- `long double std::tr1::comp_ellint_1l` (long double __k)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_2` (_Tp __k)
- `float std::tr1::comp_ellint_2f` (float __k)
- `long double std::tr1::comp_ellint_2l` (long double __k)
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::tr1::comp_ellint_3` (_Tp __k, _Tpn __nu)
- `float std::tr1::comp_ellint_3f` (float __k, float __nu)
- `long double std::tr1::comp_ellint_3l` (long double __k, long double __nu)
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type std::tr1::conf_hyperg` (_Tpa __a, _Tpc __c, _Tp __x)
- `float std::tr1::conf_hypergf` (float __a, float __c, float __x)
- `long double std::tr1::conf_hypergl` (long double __a, long double __c, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_i` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_if` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_il` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_j` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_jf` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_jl` (long double __nu, long double __x)

- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_k (_Tpnu __nu, _Tp __x)`
- `float std::tr1::cyl_bessel_kf (float __nu, float __x)`
- `long double std::tr1::cyl_bessel_kl (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_neumann (_Tpnu __nu, _Tp __x)`
- `float std::tr1::cyl_neumannf (float __nu, float __x)`
- `long double std::tr1::cyl_neumannl (long double __nu, long double __x)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_1 (_Tp __k, _Tpp __phi)`
- `float std::tr1::ellint_1f (float __k, float __phi)`
- `long double std::tr1::ellint_1l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_2 (_Tp __k, _Tpp __phi)`
- `float std::tr1::ellint_2f (float __k, float __phi)`
- `long double std::tr1::ellint_2l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpn, typename _Tpp >`
`__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type std::tr1::ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi)`
- `float std::tr1::ellint_3f (float __k, float __nu, float __phi)`
- `long double std::tr1::ellint_3l (long double __k, long double __nu, long double __phi)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::expint (_Tp __x)`
- `float std::tr1::expintf (float __x)`
- `long double std::tr1::expintl (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::fabs (_Tp __x)`
- `float std::tr1::fabs (float __x)`
- `long double std::tr1::fabs (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::hermite (unsigned int __n, _Tp __x)`
- `float std::tr1::hermitef (unsigned int __n, float __x)`
- `long double std::tr1::hermitel (unsigned int __n, long double __x)`
- `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type std::tr1::hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)`
- `float std::tr1::hypergf (float __a, float __b, float __c, float __x)`
- `long double std::tr1::hypergl (long double __a, long double __b, long double __c, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::laguerre (unsigned int __n, _Tp __x)`
- `float std::tr1::laguerref (unsigned int __n, float __x)`
- `long double std::tr1::laguerrel (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::legendre (unsigned int __n, _Tp __x)`
- `float std::tr1::legendref (unsigned int __n, float __x)`
- `long double std::tr1::legendrel (unsigned int __n, long double __x)`
- `template<typename _Tp, typename _Up >`
`__gnu_cxx::__promote_2< _Tp, _Up >::__type std::tr1::pow (_Tp __x, _Up __y)`
- `float std::tr1::pow (float __x, float __y)`
- `long double std::tr1::pow (long double __x, long double __y)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::riemann_zeta (_Tp __x)`
- `float std::tr1::riemann_zetaf (float __x)`

- long double **std::tr1::riemann_zeta** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_bessel** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_besself** (unsigned int __n, float __x)
- long double **std::tr1::sph_bessell** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_legendre** (unsigned int __l, unsigned int __m, _Tp __theta)
- float **std::tr1::sph_legendref** (unsigned int __l, unsigned int __m, float __theta)
- long double **std::tr1::sph_legendrel** (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_neumann** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_neumannf** (unsigned int __n, float __x)
- long double **std::tr1::sph_neumannl** (unsigned int __n, long double __x)

7.211.1 Detailed Description

This is a TR1 C++ Library header.

7.212 codecvr File Reference

Namespaces

- namespace [std](#)

Macros

- #define **_GLIBCXX_CODECVT**
- #define **_GLIBCXX_CODECVT_SPECIALIZATION**(_NAME, _ELEM)
- #define **_GLIBCXX_CODECVT_SPECIALIZATION2**(_NAME, _ELEM)

Enumerations

- enum **codecvr_mode** { **consume_header** , **generate_header** , **little_endian** }

7.212.1 Detailed Description

This is a Standard C++ Library header.

7.213 complex File Reference

Classes

- class [std::complex< _Tp >](#)
- class [std::complex< double >](#)
- class [std::complex< float >](#)
- class [std::complex< long double >](#)

Namespaces

- namespace [std](#)
- namespace [std::literals](#)

Macros

- #define `__cpp_lib_complex_udls`
- #define `__cpp_lib_constexpr_complex`
- #define `_GLIBCXX_COMPLEX`

Functions

- template<typename `_Tp` >
`_Tp std::__complex_abs` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_acos` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_acosh` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`_Tp std::__complex_arg` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_asin` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_asinh` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_atan` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_atanh` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_cos` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_cosh` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_exp` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_log` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_pow` (const `complex`< `_Tp` > &`__x`, const `complex`< `_Tp` > &`__y`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_pow_unsigned` (`complex`< `_Tp` > `__x`, unsigned `__n`)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::__complex_proj` (const `std::complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_sin` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_sinh` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_sqrt` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_tan` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`complex`< `_Tp` > `std::__complex_tanh` (const `complex`< `_Tp` > &`__z`)
- template<typename `_Tp` >
`_Tp std::abs` (const `complex`< `_Tp` > &)
- template<typename `_Tp` >
`std::complex`< `_Tp` > `std::acos` (const `std::complex`< `_Tp` > &`__z`)

- `template<typename _Tp >`
`std::complex< _Tp > std::acosh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::arg (_Tp __x)`
- `template<typename _Tp >`
`_Tp std::arg (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::asin (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::asinh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::atan (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::atanh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::conj (_Tp __x)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::conj (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::cos (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::cosh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::exp (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type std::imag (_Tp)`
- `template<typename _Tp >`
`constexpr _Tp std::imag (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::log (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::log10 (const complex< _Tp > &)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type std::norm (_Tp __x)`
- `template<typename _Tp >`
`_Tp constexpr std::norm (const complex< _Tp > &)`
- `template<typename _Tp >`
`constexpr _Tp std::norm (const complex< _Tp > &__z)`
- `constexpr std::complex< double > std::literals::operator""i (long double __num)`
- `constexpr std::complex< double > std::literals::operator""i (unsigned long long __num)`
- `constexpr std::complex< float > std::literals::operator""if (long double __num)`
- `constexpr std::complex< float > std::literals::operator""if (unsigned long long __num)`
- `constexpr std::complex< long double > std::literals::operator""il (long double __num)`
- `constexpr std::complex< long double > std::literals::operator""il (unsigned long long __num)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const complex< _Tp > &__x)`

- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, complex< _Tp > &__x)`
- `template<typename _Tp >`
`complex< _Tp > std::polar (const _Tp &, const _Tp &=0)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const _Tp &, const complex< _Tp > &)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const _Tp &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, int)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const std::complex< _Tp > &__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const std::complex< _Tp > &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::proj (_Tp __x)`
- `template<typename _Tp >`
`std::complex< _Tp > std::proj (const std::complex< _Tp > &)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__promote< _Tp >::__type std::real (_Tp __x)`
- `template<typename _Tp >`
`constexpr _Tp std::real (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::sin (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sinh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sqrt (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tan (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tanh (const complex< _Tp > &)`

- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator+ (const complex< _Tp > &__x, const complex< _Tp > &__y)`

- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const _Tp &__x, const complex< _Tp > &__y)`

- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator- (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator* (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr complex< _Tp > std::operator/ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const complex< _Tp > &__y)`

7.213.1 Detailed Description

This is a Standard C++ Library header.

7.214 complex File Reference

Namespaces

- namespace [std](#)
- namespace [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_COMPLEX`

Functions

- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::tr1::conj (_Tp __x)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::conj (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::polar (const _Tp &__rho, const _Up &__theta)`

- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const _Tp &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex< _Tp > &__x, const _Up &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex< _Tp > &__x, const std::complex< _Up > &__y)`

7.214.1 Detailed Description

This is a TR1 C++ Library header.

7.215 complex.h File Reference

Macros

- `#define _GLIBCXX_COMPLEX_H`

7.215.1 Detailed Description

This is a Standard C++ Library header.

7.216 Concepts

Here is a list of all documented concepts with brief descriptions:

<code>std::assignable_from</code>	
[concept.assignable], concept assignable_from	791
<code>std::common_reference_with</code>	
[concept.commonref], concept common_reference_with	791
<code>std::common_with</code>	
[concept.common], concept common_with	791
<code>std::constructible_from</code>	
[concept.constructible], concept constructible_from	791
<code>std::convertible_to</code>	
[concept.convertible], concept convertible_to	792
<code>std::copy_constructible</code>	
[concept.copyconstructible], concept copy_constructible	792
<code>std::default_initializable</code>	
[concept.defaultinitializable], concept default_initializable	792

<code>std::derived_from</code>	
[concept.derived], concept <code>derived_from</code>	792
<code>std::destructible</code>	
[concept.destructible], concept <code>destructible</code>	792
<code>std::equivalence_relation</code>	
[concept.equiv], concept <code>equivalence_relation</code>	793
<code>std::indirectly_comparable</code>	
[alg.req.ind.cmp], concept <code>indirectly_comparable</code>	793
<code>std::indirectly_copyable</code>	
[alg.req.ind.copy], concept <code>indirectly_copyable</code>	793
<code>std::indirectly_movable</code>	
[alg.req.ind.move], concept <code>indirectly_movable</code>	793
<code>std::indirectly_readable</code>	
Requirements for types that are readable by applying operator*	793
<code>std::indirectly_swappable</code>	
[alg.req.ind.swap], concept <code>indirectly_swappable</code>	794
<code>std::indirectly_writable</code>	
Requirements for writing a value into an iterator's referenced object	794
<code>std::invocable</code>	
[concept.invocable], concept <code>invocable</code>	794
<code>std::mergeable</code>	
[alg.req.mergeable], concept <code>mergeable</code>	794
<code>std::move_constructible</code>	
[concept.moveconstructible], concept <code>move_constructible</code>	795
<code>std::permutable</code>	
[alg.req.permutable], concept <code>permutable</code>	795
<code>std::predicate</code>	
[concept.predicate], concept <code>predicate</code>	795
<code>std::ranges::bidirectional_range</code>	
A range for which <code>ranges::begin</code> returns a bidirectional iterator	795
<code>std::ranges::borrowed_range</code>	
[range.range] The <code>borrowed_range</code> concept	796
<code>std::ranges::common_range</code>	
A range for which <code>ranges::begin</code> and <code>ranges::end</code> return the same type	796
<code>std::ranges::contiguous_range</code>	
A range for which <code>ranges::begin</code> returns a contiguous iterator	796
<code>std::ranges::forward_range</code>	
A range for which <code>ranges::begin</code> returns a forward iterator	796

<code>std::ranges::input_range</code>	
A range for which <code>ranges::begin</code> returns an input iterator	797
<code>std::ranges::output_range</code>	
A range for which <code>ranges::begin</code> returns an output iterator	797
<code>std::ranges::random_access_range</code>	
A range for which <code>ranges::begin</code> returns a random access iterator	797
<code>std::ranges::range</code>	
[<code>range.range</code>] The range concept	797
<code>std::ranges::sized_range</code>	
[<code>range.sized</code>] The <code>sized_range</code> concept	797
<code>std::ranges::view</code>	
[<code>range.view</code>] The <code>ranges::view</code> concept	798
<code>std::ranges::viewable_range</code>	
A range which can be safely converted to a view	798
<code>std::regular_invocable</code>	
[<code>concept.regularinvocable</code>], concept <code>regular_invocable</code>	798
<code>std::relation</code>	
[<code>concept.relation</code>], concept <code>relation</code>	798
<code>std::same_as</code>	
[<code>concept.same</code>], concept <code>same_as</code>	799
<code>std::sortable</code>	
[<code>alg.req.sortable</code>], concept <code>sortable</code>	799
<code>std::strict_weak_order</code>	
[<code>concept.strictweakorder</code>], concept <code>strict_weak_order</code>	799
<code>std::uniform_random_bit_generator</code>	
Requirements for a uniform random bit generator	799
<code>std::weakly_incrementable</code>	
Requirements on types that can be incremented with <code>++</code>	799

7.217 `condition_variable` File Reference

Classes

- class [`std::condition_variable`](#)
- class [`std::_V2::condition_variable_any`](#)

Namespaces

- namespace [`std`](#)

Macros

- `#define` [`_GLIBCXX_CONDITION_VARIABLE`](#)

Enumerations

- enum class `std::cv_status` { `no_timeout` , `timeout` }

Functions

- void `std::notify_all_at_thread_exit` (`condition_variable` &, `unique_lock`< `mutex` >)

7.217.1 Detailed Description

This is a Standard C++ Library header.

7.218 csetjmp File Reference

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_CSETJMP`
- `#define setjmp(env)`

7.218.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `setjmp.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.219 csignal File Reference

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_CSIGNAL`

7.219.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `signal.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.220 cstdalign File Reference

Macros

- `#define _GLIBCXX_CSTDALIGN`

7.220.1 Detailed Description

This is a Standard C++ Library header.

7.221 cstdarg File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CSTDARG`
- `#define va_end(ap)`

7.221.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdarg.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.222 cstdarg File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDARG`

7.222.1 Detailed Description

This is a TR1 C++ Library header.

7.223 cstdlib File Reference

Macros

- `#define _GLIBCXX_CSTDBOOL`

7.223.1 Detailed Description

This is a Standard C++ Library header.

7.224 cstdlib File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDBOOL`

7.224.1 Detailed Description

This is a TR1 C++ Library header.

7.225 cstddef File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_byte`
- `#define _GLIBCXX_CSTDDEF`

Typedefs

- `template<typename _IntegerType >`
`using std::__byte_op_t = typename __byte_operand< _IntegerType >::__type`

Enumerations

- enum class `std::byte` : unsigned char

Functions

- `constexpr byte std::operator&(byte __l, byte __r) noexcept`
- `constexpr byte & std::operator&=(byte &__l, byte __r) noexcept`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > std::operator<<(byte __b, _IntegerType __shift) noexcept`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > & std::operator<<=(byte &__b, _IntegerType __shift) noexcept`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > std::operator>>(byte __b, _IntegerType __shift) noexcept`
- `template<typename _IntegerType >`
`constexpr __byte_op_t< _IntegerType > & std::operator>>=(byte &__b, _IntegerType __shift) noexcept`
- `constexpr byte std::operator^(byte __l, byte __r) noexcept`
- `constexpr byte & std::operator^=(byte &__l, byte __r) noexcept`
- `constexpr byte std::operator|(byte __l, byte __r) noexcept`
- `constexpr byte & std::operator|=(byte &__l, byte __r) noexcept`
- `constexpr byte std::operator~(byte __b) noexcept`
- `template<typename _IntegerType >`
`constexpr _IntegerType std::to_integer(__byte_op_t< _IntegerType > __b) noexcept`

7.225.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdint.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.226 cstdint File Reference

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_CSTDINT`

7.226.1 Detailed Description

This is a Standard C++ Library header.

7.227 cstdint File Reference

Namespaces

- namespace `std`
- namespace `std::tr1`

Macros

- `#define _GLIBCXX_TR1_CSTDINT`

7.227.1 Detailed Description

This is a TR1 C++ Library header.

7.228 cstdio File Reference**Namespaces**

- namespace [std](#)

Macros

- `#define _GLIBCXX_CSTDIO`

7.228.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdio.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.229 cstdlib File Reference**Macros**

- `#define _GLIBCXX_TR1_CSTDIO`

7.229.1 Detailed Description

This is a TR1 C++ Library header.

7.230 cstdlib File Reference**Namespaces**

- namespace [std](#)

Macros

- `#define _GLIBCXX_CSTDLIB`
- `#define _GLIBCXX_INCLUDE_NEXT_C_HEADERS`

Functions

- `ldiv_t std::div (long __i, long __j)`

7.230.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdlib.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.231 cstdlib File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDLIB`

7.231.1 Detailed Description

This is a TR1 C++ Library header.

7.232 cstring File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CSTRING`

Functions

- `void * std::memchr (void *__s, int __c, size_t __n)`
- `char * std::strchr (char *__s, int __n)`
- `char * std::strpbrk (char *__s1, const char *__s2)`
- `char * std::strrchr (char *__s, int __n)`
- `char * std::strstr (char *__s1, const char *__s2)`

7.232.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `string.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.233 ctgmath File Reference

Macros

- `#define _GLIBCXX_CTGMATH`

7.233.1 Detailed Description

This is a Standard C++ Library header.

7.234 ctgmath File Reference

Macros

- `#define _GLIBCXX_TR1_CTGMATH`

7.234.1 Detailed Description

This is a TR1 C++ Library header.

7.235 ctime File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CTIME`

7.235.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `time.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.236 ctime File Reference

Macros

- `#define _GLIBCXX_TR1_CTIME`

7.236.1 Detailed Description

This is a TR1 C++ Library header.

7.237 cuchar File Reference

Macros

- `#define _GLIBCXX_CUCHAR`

7.237.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `uchar.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.238 wchar File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_CWCHAR`

Functions

- `wchar_t * std::wcschr (wchar_t *__p, wchar_t __c)`
- `wchar_t * std::wcpbrk (wchar_t *__s1, const wchar_t *__s2)`
- `wchar_t * std::wcsrchr (wchar_t *__p, wchar_t __c)`
- `wchar_t * std::wcsstr (wchar_t *__s1, const wchar_t *__s2)`
- `wchar_t * std::wmemchr (wchar_t *__p, wchar_t __c, size_t __n)`

7.238.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `wchar.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.239 `cwchar` File Reference

Namespaces

- namespace `std`
- namespace `std::tr1`

Macros

- `#define _GLIBCXX_TR1_CWCHAR`

7.239.1 Detailed Description

This is a TR1 C++ Library header.

7.240 `cwctype` File Reference

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_CWCTYPE`

7.240.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `wctype.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

7.241 `cwctype` File Reference

Namespaces

- namespace `std`
- namespace `std::tr1`

Macros

- `#define _GLIBCXX_TR1_CWCTYPE`

7.241.1 Detailed Description

This is a TR1 C++ Library header.

7.242 assertions.h File Reference

Macros

- `#define __glibcxx_requires_non_empty_range(_First, _Last)`
- `#define __glibcxx_requires_nonempty()`
- `#define __glibcxx_requires_subscript(_N)`
- `#define _GLIBCXX_DEBUG_ASSERT(_Condition)`
- `#define _GLIBCXX_DEBUG_ONLY(_Statement)`
- `#define _GLIBCXX_DEBUG_PEDASSERT(_Condition)`

7.242.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.243 debug.h File Reference

Namespaces

- namespace [__gnu_debug](#)
- namespace [std](#)
- namespace [std::__debug](#)

Macros

- `#define __glibcxx_requires_can_decrement_range(_First1, _Last1, _First2)`
- `#define __glibcxx_requires_can_increment(_First, _Size)`
- `#define __glibcxx_requires_can_increment_range(_First1, _Last1, _First2)`
- `#define __glibcxx_requires_cond(_Cond, _Msg)`
- `#define __glibcxx_requires_heap(_First, _Last)`
- `#define __glibcxx_requires_heap_pred(_First, _Last, _Pred)`
- `#define __glibcxx_requires_irreflexive(_First, _Last)`
- `#define __glibcxx_requires_irreflexive2(_First, _Last)`
- `#define __glibcxx_requires_irreflexive_pred(_First, _Last, _Pred)`
- `#define __glibcxx_requires_irreflexive_pred2(_First, _Last, _Pred)`
- `#define __glibcxx_requires_partitioned_lower(_First, _Last, _Value)`
- `#define __glibcxx_requires_partitioned_lower_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_requires_partitioned_upper(_First, _Last, _Value)`
- `#define __glibcxx_requires_partitioned_upper_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_requires_sorted(_First, _Last)`
- `#define __glibcxx_requires_sorted_pred(_First, _Last, _Pred)`
- `#define __glibcxx_requires_sorted_set(_First1, _Last1, _First2)`
- `#define __glibcxx_requires_sorted_set_pred(_First1, _Last1, _First2, _Pred)`
- `#define __glibcxx_requires_string(_String)`
- `#define __glibcxx_requires_string_len(_String, _Len)`
- `#define __glibcxx_requires_valid_range(_First, _Last)`

7.243.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.244 formatter.h File Reference

Namespaces

- namespace [__gnu_cxx](#)
- namespace [__gnu_debug](#)
- namespace [std](#)

Macros

- `#define _GLIBCXX_TYPEID(_Type)`

Enumerations

- enum `_Debug_msg_id` {
`__msg_valid_range`, `__msg_insert_singular`, `__msg_insert_different`, `__msg_erase_bad`,
`__msg_erase_different`, `__msg_subscript_oob`, `__msg_empty`, `__msg_unpartitioned`,
`__msg_unpartitioned_pred`, `__msg_unsorted`, `__msg_unsorted_pred`, `__msg_not_heap`,
`__msg_not_heap_pred`, `__msg_bad_bitset_write`, `__msg_bad_bitset_read`, `__msg_bad_bitset_flip`,
`__msg_self_splice`, `__msg_splice_alloc`, `__msg_splice_bad`, `__msg_splice_other`,
`__msg_splice_overlap`, `__msg_init_singular`, `__msg_init_copy_singular`, `__msg_init_const_singular`,
`__msg_copy_singular`, `__msg_bad_deref`, `__msg_bad_inc`, `__msg_bad_dec`,
`__msg_iter_subscript_oob`, `__msg_advance_oob`, `__msg_retreat_oob`, `__msg_iter_compare_bad`,
`__msg_compare_different`, `__msg_iter_order_bad`, `__msg_order_different`, `__msg_distance_bad`,
`__msg_distance_different`, `__msg_deref_istream`, `__msg_inc_istream`, `__msg_output_ostream`,
`__msg_deref_istreambuf`, `__msg_inc_istreambuf`, `__msg_insert_after_end`, `__msg_erase_after_bad`,
`__msg_valid_range2`, `__msg_local_iter_compare_bad`, `__msg_non_empty_range`, `__msg_self_move`↵
`__assign`,
`__msg_bucket_index_oob`, `__msg_valid_load_factor`, `__msg_equal_allocs`, `__msg_insert_range`↵
`__from_self`,
`__msg_irreflexive_ordering` }

Functions

- `template<typename _Iterator >`
`constexpr bool __gnu_debug::__check_singular (_Iterator const &)`

7.244.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.245 functions.h File Reference

Namespaces

- namespace [__gnu_debug](#)

Functions

- `template<typename _ForwardIterator, typename _Tp >`
`constexpr bool __gnu_debug::__check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`constexpr bool __gnu_debug::__check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__value, _Pred __pred)`

- `template<typename _ForwardIterator, typename _Tp >`
`constexpr bool __gnu_debug::__check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`constexpr bool __gnu_debug::__check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__value, _Pred __pred)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::__check_sorted (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted (const _InputIterator &__first, const _InputIterator &__last, _↵`
`Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, _↵`
`Predicate __pred, std::forward_iterator_tag)`
- `template<typename _ForwardIterator >`
`constexpr bool __gnu_debug::__check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last,`
`std::forward_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted_aux (const _InputIterator &, const _InputIterator &, _Predicate,`
`std::input_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::__check_sorted_aux (const _InputIterator &, const _InputIterator &,`
`std::input_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`constexpr bool __gnu_debug::__check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__↵`
`last, const _InputIterator2 &)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__↵`
`last, const _InputIterator2 &, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &, const _InputIterator &, _↵`
`Predicate, std::false_type)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &, const _InputIterator &, std::↵`
`false_type)`
- `template<typename _InputIterator, typename _Predicate >`
`constexpr bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &↵`
`__last, _Predicate __pred, std::true_type)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &↵`
`__last, std::true_type)`
- `template<typename _InputIterator >`
`_InputIterator __gnu_debug::__check_valid_range (const _InputIterator &__first, const _InputIterator &__last,`
`const char *__file, unsigned int __line, const char *__function)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator (const _Safe_iterator< _Iterator, _Sequence, _Category > &__it, _↵`
`_InputIterator __other, _InputIterator __other_end)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Integral >`
`bool __gnu_debug::__foreign_iterator_aux (const _Safe_iterator< _Iterator, _Sequence, _Category > &, _↵`
`_Integral, _Integral, std::true_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux (const _Safe_iterator< _Iterator, _Sequence, _Category > &__it,`
`_InputIterator __other, _InputIterator __other_end, std::false_type)`

- `template<typename _Iterator, typename _Sequence, typename _Category, typename _OtherIterator, typename _OtherSequence, typename _OtherCategory >`
`bool __gnu_debug::__foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &, const __Safe_iterator< _OtherIterator, _OtherSequence, _OtherCategory > &, const __Safe_iterator< _OtherIterator, _OtherSequence, _OtherCategory > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const _InputIterator &__other, const _InputIterator &__other_end)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _OtherIterator >`
`bool __gnu_debug::__foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const __Safe_iterator< _OtherIterator, _Sequence, _Category > &__other, const __Safe_iterator< _OtherIterator, _Sequence, _Category > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence, _Category > &, const _InputIterator &, const _InputIterator &, std::false_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const _InputIterator &__other, const _InputIterator &__other_end, std::true_type)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __gnu_debug::__foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence, _Category > &,...)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __gnu_debug::__foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence, _Category > &__it, const typename _Sequence::value_type *__other)`
- `template<typename _Iterator >`
`constexpr bool __gnu_debug::__is_irreflexive (_Iterator __it)`
- `template<typename _Iterator, typename _Pred >`
`constexpr bool __gnu_debug::__is_irreflexive_pred (_Iterator __it, _Pred __pred)`

7.245.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.246 helper_functions.h File Reference

Namespaces

- namespace `__gnu_debug`

Enumerations

- enum `__gnu_debug::__Distance_precision` {
`__dp_none`, `__dp_equality`, `__dp_sign`, `__dp_sign_max_size`,
`__dp_exact` }

Functions

- `template<typename _Iterator >`
`constexpr _Iterator __gnu_debug::__base (_Iterator __it)`
- `template<typename _InputIterator, typename _Size >`
`constexpr bool __gnu_debug::__can_advance (_InputIterator, _Size)`
- `template<typename _InputIterator, typename _Diff >`
`constexpr bool __gnu_debug::__can_advance (_InputIterator, const std::pair< _Diff, __Distance_precision > &, int)`

- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Size >`
`bool __gnu_debug::can_advance (const _Safe_iterator< _Iterator, _Sequence, _Category > &, _Size)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Diff >`
`bool __gnu_debug::can_advance (const _Safe_iterator< _Iterator, _Sequence, _Category > &, const std::pair< _Diff, _Distance_precision > &, int)`
- `template<typename _Iterator >`
`constexpr bool __gnu_debug::check_singular (_Iterator const &)`
- `template<typename _Tp >`
`constexpr bool __gnu_debug::check_singular (_Tp *const &__ptr)`
- `bool __gnu_debug::check_singular_aux (const void *)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __gnu_debug::get_distance (_Iterator __lhs, _Iterator __↵
rhs)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __gnu_debug::get_distance (_Iterator __lhs, _Iterator __↵
rhs, std::input_iterator_tag)`
- `template<typename _Iterator >`
`constexpr _Distance_traits< _Iterator >::__type __gnu_debug::get_distance (_Iterator __lhs, _Iterator __rhs,
std::random_access_iterator_tag)`
- `template<typename _Iterator >`
`_Iterator __gnu_debug::unsafe (_Iterator __it)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range (_InputIterator __first, _InputIterator __last, typename _Distance↵
_traits< _InputIterator >::__type &__dist)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __gnu_debug::valid_range (const _Safe_iterator< _Iterator, _Sequence, _Category > &, const
_Safe_iterator< _Iterator, _Sequence, _Category > &)`
- `template<typename _Iterator, typename _Sequence, typename _Category >`
`bool __gnu_debug::valid_range (const _Safe_iterator< _Iterator, _Sequence, _Category > &, const
_Safe_iterator< _Iterator, _Sequence, _Category > &, typename _Distance_traits< _Iterator >::__type &)`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator<
_Iterator, _Sequence > &)`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator<
_Iterator, _Sequence > &, typename _Distance_traits< _Iterator >::__type &)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range_aux (_InputIterator __first, _InputIterator __last, std::false_type)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range_aux (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range_aux (_InputIterator __first, _InputIterator __last, std::random_access_iterator_tag)`
- `template<typename _InputIterator >`
`constexpr bool __gnu_debug::valid_range_aux (_InputIterator __first, _InputIterator __last, typename ↵
_Distance_traits< _InputIterator >::__type &__dist, std::false_type)`
- `template<typename _Integral >`
`constexpr bool __gnu_debug::valid_range_aux (_Integral, _Integral, std::true_type)`
- `template<typename _Integral >`
`constexpr bool __gnu_debug::valid_range_aux (_Integral, _Integral, typename _Distance_traits< _Integral
>::__type &__dist, std::true_type)`

7.246.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.247 macros.h File Reference

Macros

- `#define __glibcxx_check_bucket_index(_N)`
- `#define __glibcxx_check_can_decrement_range(_First1, _Last1, _First2)`
- `#define __glibcxx_check_can_increment(_First, _Size)`
- `#define __glibcxx_check_can_increment_dist(_First, _Dist, _Way)`
- `#define __glibcxx_check_can_increment_range(_First1, _Last1, _First2)`
- `#define __glibcxx_check_equal_allocs(_This, _Other)`
- `#define __glibcxx_check_erase(_Position)`
- `#define __glibcxx_check_erase2(_CPosition)`
- `#define __glibcxx_check_erase_after(_Position)`
- `#define __glibcxx_check_erase_range(_First, _Last)`
- `#define __glibcxx_check_erase_range_after(_First, _Last)`
- `#define __glibcxx_check_heap(_First, _Last)`
- `#define __glibcxx_check_heap_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_insert(_Position)`
- `#define __glibcxx_check_insert_after(_Position)`
- `#define __glibcxx_check_insert_range(_Position, _First, _Last, _Dist)`
- `#define __glibcxx_check_insert_range_after(_Position, _First, _Last, _Dist)`
- `#define __glibcxx_check_irreflexive(_First, _Last)`
- `#define __glibcxx_check_irreflexive2(_First, _Last)`
- `#define __glibcxx_check_irreflexive_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_irreflexive_pred2(_First, _Last, _Pred)`
- `#define __glibcxx_check_max_load_factor(_F)`
- `#define __glibcxx_check_non_empty_range(_First, _Last)`
- `#define __glibcxx_check_nonempty()`
- `#define __glibcxx_check_partitioned_lower(_First, _Last, _Value)`
- `#define __glibcxx_check_partitioned_lower_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_check_partitioned_upper(_First, _Last, _Value)`
- `#define __glibcxx_check_partitioned_upper_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_check_sorted(_First, _Last)`
- `#define __glibcxx_check_sorted_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_sorted_set(_First1, _Last1, _First2)`
- `#define __glibcxx_check_sorted_set_pred(_First1, _Last1, _First2, _Pred)`
- `#define __glibcxx_check_string(_String)`
- `#define __glibcxx_check_string_len(_String, _Len)`
- `#define __glibcxx_check_subscript(_N)`
- `#define __glibcxx_check_valid_constructor_range(_First, _Last)`
- `#define __glibcxx_check_valid_range(_First, _Last)`
- `#define __glibcxx_check_valid_range2(_First, _Last, _Dist)`
- `#define __glibcxx_check_valid_range_at(_First, _Last, _File, _Line, _Func)`
- `#define _GLIBCXX_DEBUG_VERIFY(_Cond, _ErrMsg)`
- `#define _GLIBCXX_DEBUG_VERIFY_AT(_Cond, _ErrMsg, _File, _Line)`
- `#define _GLIBCXX_DEBUG_VERIFY_AT_F(_Cond, _ErrMsg, _File, _Line, _Func)`
- `#define _GLIBCXX_DEBUG_VERIFY_COND_AT(_Cond, _ErrMsg, _File, _Line, _Func)`

7.247.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.247.2 Macro Definition Documentation

`__glibcxx_check_erase`

```
#define __glibcxx_check_erase(  
    _Position )
```

Verify that we can erase the element referenced by the iterator `_Position`. We can erase the element if the `_Position` iterator is dereferenceable and references this sequence.

`__glibcxx_check_erase_after`

```
#define __glibcxx_check_erase_after(  
    _Position )
```

Verify that we can erase the element after the iterator `_Position`. We can erase the element if the `_Position` iterator is before a dereferenceable one and references this sequence.

`__glibcxx_check_erase_range`

```
#define __glibcxx_check_erase_range(  
    _First,  
    _Last )
```

Verify that we can erase the elements in the iterator range `[_First, _Last)`. We can erase the elements if `[_First, _Last)` is a valid iterator range within this sequence.

`__glibcxx_check_erase_range_after`

```
#define __glibcxx_check_erase_range_after(  
    _First,  
    _Last )
```

Verify that we can erase the elements in the iterator range `(_First, _Last)`. We can erase the elements if `(_First, _Last)` is a valid iterator range within this sequence.

`__glibcxx_check_heap_pred`

```
#define __glibcxx_check_heap_pred(  
    _First,  
    _Last,  
    _Pred )
```

Verify that the iterator range `[_First, _Last)` is a heap w.r.t. the predicate `_Pred`.

`__glibcxx_check_insert`

```
#define __glibcxx_check_insert(  
    _Position )
```

Verify that we can insert into `*this` with the iterator `_Position`. Insertion into a container at a specific position requires that the iterator be nonsingular, either dereferenceable or past-the-end, and that it reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the iterator is a `_Safe_iterator`.

`__glibcxx_check_insert_after`

```
#define __glibcxx_check_insert_after(  
    _Position )
```

Verify that we can insert into **this* after the iterator `_Position`. Insertion into a container after a specific position requires that the iterator be nonsingular, either dereferenceable or before-begin, and that it reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the iterator is a `_Safe_↔` iterator.

`__glibcxx_check_insert_range`

```
#define __glibcxx_check_insert_range(
    _Position,
    _First,
    _Last,
    _Dist )
```

Verify that we can insert the values in the iterator range `[_First, _Last)` into **this* with the iterator `_Position`. Insertion into a container at a specific position requires that the iterator be nonsingular (i.e., either dereferenceable or past-the-end), that it reference the sequence we are inserting into, and that the iterator range `[_First, _Last)` is a valid (possibly empty) range which does not reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the `_Position` iterator is a `_Safe_iterator`.

`__glibcxx_check_insert_range_after`

```
#define __glibcxx_check_insert_range_after(
    _Position,
    _First,
    _Last,
    _Dist )
```

Verify that we can insert the values in the iterator range `[_First, _Last)` into **this* after the iterator `_Position`. Insertion into a container after a specific position requires that the iterator be nonsingular (i.e., either dereferenceable or past-the-end), that it reference the sequence we are inserting into, and that the iterator range `[_First, _Last)` is a valid (possibly empty) range which does not reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the `_Position` iterator is a `_Safe_iterator`.

`__glibcxx_check_partitioned_lower`

```
#define __glibcxx_check_partitioned_lower(
    _First,
    _Last,
    _Value )
```

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value`.

`__glibcxx_check_partitioned_lower_pred`

```
#define __glibcxx_check_partitioned_lower_pred(
    _First,
    _Last,
    _Value,
    _Pred )
```

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value` and predicate `_Pred`.

`__glibcxx_check_partitioned_upper_pred`

```
#define __glibcxx_check_partitioned_upper_pred(
    _First,
    _Last,
    _Value,
    _Pred )
```

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value` and predicate `_Pred`.

`__glibcxx_check_sorted_pred`

```
#define __glibcxx_check_sorted_pred(
    _First,
    _Last,
    _Pred )
```

Verify that the iterator range `[_First, _Last)` is sorted by the predicate `_Pred`.

`_GLIBCXX_DEBUG_VERIFY_COND_AT`

```
#define _GLIBCXX_DEBUG_VERIFY_COND_AT(
    _Cond,
    _ErrMsg,
    _File,
    _Line,
    _Func )
```

Macros used by the implementation to verify certain properties. These macros may only be used directly by the debug wrappers. Note that these are macros (instead of the more obviously *correct* choice of making them functions) because we need line and file information at the call site, to minimize the distance between the user error and where the error is reported.

7.248 map.h File Reference

Classes

- class `std::__debug::map<_Key, _Tp, _Compare, _Allocator>`

Namespaces

- namespace `std`
- namespace `std::__debug`

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::map` (`_InputIterator`, `_InputIterator`, `_Allocator`) -> `map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, less<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::map` (`_InputIterator`, `_InputIterator`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Compare, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::__debug::map` (`initializer_list<pair<_Key, _Tp>>`, `_Allocator`) -> `map<_Key, _Tp, less<_Key>, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::map` (`initializer_list<pair<_Key, _Tp>>`, `_Compare=_Compare()`, `_Allocator=_Allocator()`) -> `map<_Key, _Tp, _Compare, _Allocator>`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator>`
`bool std::__debug::operator!=` (`const map<_Key, _Tp, _Compare, _Allocator> &__lhs`, `const map<_Key, _Tp, _Compare, _Allocator> &__rhs`)

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void std::__debug::swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`

7.248.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.249 multimap.h File Reference

Classes

- class `std::__debug::multimap< _Key, _Tp, _Compare, _Allocator >`

Namespaces

- namespace `std`
- namespace `std::__debug`

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::multimap (_InputIterator, _InputIterator, _Allocator) -> multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, less< __iter_key_t< _InputIterator > >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less< __iter_key_t< _InputIterator > >, typename _Allocator = allocator< __iter_to_alloc_t< _InputIterator > >, typename = _RequireInputIter< _InputIterator >, typename = _RequireNotAllocator< _Compare >, typename = _RequireAllocator< _Allocator >>`
`std::__debug::multimap (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Compare, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator< _Allocator >>`
`std::__debug::multimap (initializer_list< pair< _Key, _Tp > >, _Allocator) -> multimap< _Key, _Tp, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare = less< _Key >, typename _Allocator = allocator< pair< const _Key, _Tp > >, typename = _RequireNotAllocator< _Compare >, typename = _RequireAllocator< _Allocator >>`
`std::__debug::multimap (initializer_list< pair< _Key, _Tp > >, _Compare=_Compare(), _Allocator=_Allocator()) -> multimap< _Key, _Tp, _Compare, _Allocator >`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator!= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void std::__debug::swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _`
`Compare, _Allocator > &__rhs) noexcept(/*conditional */)`

7.249.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.250 [multiset.h](#) File Reference

Classes

- class [std::__debug::multiset](#)< [_Key](#), [_Compare](#), [_Allocator](#) >

Namespaces

- namespace [std](#)
- namespace [std::__debug](#)

Functions

- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllo-`
`Allocator<_Allocator>>`
`std::__debug::multiset (_InputIterator, _InputIterator, _Allocator) -> multiset< typename iterator_traits< _
InputIterator >::value_type, less< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`
- `template<typename _InputIterator, typename _Compare = less<typename iterator_traits<_InputIterator>::value_type>, typename _`
`Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::multiset (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) ->`
`multiset< typename iterator_traits< _InputIterator >::value_type, _Compare, _Allocator >`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::__debug::multiset (initializer_list< _Key >, _Allocator) -> multiset< _Key, less< _Key >, _Allocator >`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNot`
`Allocator<_Compare>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::multiset (initializer_list< _Key >, _Compare=_Compare(), _Allocator=_Allocator()) -> multiset<`
`_Key, _Compare, _Allocator >`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator!= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`

- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void std::__debug::swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`

7.250.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.251 `safe_base.h` File Reference

Classes

- class `__gnu_debug::__Safe_iterator_base`
- class `__gnu_debug::__Safe_sequence_base`

Namespaces

- namespace `__gnu_debug`

Functions

- `bool __gnu_debug::__check_singular_aux (const __Safe_iterator_base *__x)`

7.251.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.252 `safe_container.h` File Reference

Classes

- class `__gnu_debug::__Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >`

Namespaces

- namespace `__gnu_debug`

7.252.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.253 `safe_iterator.h` File Reference

Classes

- struct `__gnu_debug::BeforeBeginHelper<_Sequence>`
- class `__gnu_debug::Safe_iterator<_Iterator, _Sequence, _Category>`
- struct `__gnu_debug::Sequence_traits<_Sequence>`

Namespaces

- namespace `__gnu_debug`

Macros

- `#define _GLIBCXX_DEBUG_VERIFY_DIST_OPERANDS(_Lhs, _Rhs)`
- `#define _GLIBCXX_DEBUG_VERIFY_EQ_OPERANDS(_Lhs, _Rhs)`
- `#define _GLIBCXX_DEBUG_VERIFY_OPERANDS(_Lhs, _Rhs, _BadMsgId, _DiffMsgId)`
- `#define _GLIBCXX_DEBUG_VERIFY_REL_OPERANDS(_Lhs, _Rhs)`

Functions

- `template<typename _Iterator, typename _Sequence>`
`_Iterator __gnu_debug::__base (const _Safe_iterator<_Iterator, _Sequence, std::random_access_iterator_tag> & __it)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Size>`
`bool __gnu_debug::__can_advance (const _Safe_iterator<_Iterator, _Sequence, _Category> &, _Size)`
- `template<typename _Iterator, typename _Sequence, typename _Category, typename _Diff>`
`bool __gnu_debug::__can_advance (const _Safe_iterator<_Iterator, _Sequence, _Category> &, const std::pair<_Diff, _Distance_precision> &, int)`
- `template<typename _Iterator, typename _Sequence>`
`_Iterator __gnu_debug::__unsafe (const _Safe_iterator<_Iterator, _Sequence> & __it)`
- `template<typename _Iterator, typename _Sequence, typename _Category>`
`bool __gnu_debug::__valid_range (const _Safe_iterator<_Iterator, _Sequence, _Category> &, const _Safe_iterator<_Iterator, _Sequence, _Category> &)`
- `template<typename _Iterator, typename _Sequence, typename _Category>`
`bool __gnu_debug::__valid_range (const _Safe_iterator<_Iterator, _Sequence, _Category> &, const _Safe_iterator<_Iterator, _Sequence, _Category> &, typename _Distance_traits<_Iterator>::__type &)`

7.253.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.254 `safe_iterator.tcc` File Reference

Namespaces

- namespace `__gnu_debug`
- namespace `std`

Macros

- `#define _GLIBCXX_DEBUG_SAFE_ITERATOR_TCC`

Functions

- `template<bool _IsMove, typename _II, typename _Ite, typename _Seq, typename _Cat >`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > std::__copy_move_a (_II, _II, const __gnu_debug::Safe_iterator<`
`_Ite, _Seq, _Cat > &)`
- `template<bool _IsMove, typename _IIte, typename _ISeq, typename _ICat, typename _OIte, typename _OSeq, typename _OCat >`
`__gnu_debug::Safe_iterator< _OIte, _OSeq, _OCat > std::__copy_move_a (const __gnu_debug::Safe_iterator<`
`_IIte, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator< _IIte, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator<`
`_OIte, _OSeq, _OCat > &)`
- `template<bool _IsMove, typename _Ite, typename _Seq, typename _Cat, typename _OI >`
`_OI std::__copy_move_a (const __gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &, const __gnu_debug::Safe_iterator<`
`_Ite, _Seq, _Cat > &, _OI)`
- `template<bool _IsMove, typename _II, typename _Ite, typename _Seq, typename _Cat >`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > std::__copy_move_backward_a (_II, _II, const`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &)`
- `template<bool _IsMove, typename _IIte, typename _ISeq, typename _ICat, typename _OIte, typename _OSeq, typename _OCat >`
`__gnu_debug::Safe_iterator< _OIte, _OSeq, _OCat > std::__copy_move_backward_a (const __gnu_debug::Safe_iterator<`
`_IIte, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator< _IIte, _ISeq, _ICat > &, const __gnu_debug::Safe_iterator<`
`_OIte, _OSeq, _OCat > &)`
- `template<bool _IsMove, typename _Ite, typename _Seq, typename _Cat, typename _OI >`
`_OI std::__copy_move_backward_a (const __gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &, const`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &, _OI)`
- `template<typename _II1, typename _II2, typename _Seq2, typename _Cat2 >`
`bool std::__equal_aux (_II1, _II1, const __gnu_debug::Safe_iterator< _II2, _Seq2, _Cat2 > &)`
- `template<typename _II1, typename _Seq1, typename _Cat1, typename _II2 >`
`bool std::__equal_aux (const __gnu_debug::Safe_iterator< _II1, _Seq1, _Cat1 > &, const __gnu_debug::Safe_iterator<`
`_II1, _Seq1, _Cat1 > &, _II2)`
- `template<typename _II1, typename _Seq1, typename _Cat1, typename _II2, typename _Seq2, typename _Cat2 >`
`bool std::__equal_aux (const __gnu_debug::Safe_iterator< _II1, _Seq1, _Cat1 > &, const __gnu_debug::Safe_iterator<`
`_II1, _Seq1, _Cat1 > &, const __gnu_debug::Safe_iterator< _II2, _Seq2, _Cat2 > &)`
- `template<typename _Ite, typename _Seq, typename _Cat, typename _Tp >`
`void std::__fill_a (const __gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > &, const __gnu_debug::Safe_iterator<`
`_Ite, _Seq, _Cat > &, const _Tp &)`
- `template<typename _Ite, typename _Seq, typename _Cat, typename _Size, typename _Tp >`
`__gnu_debug::Safe_iterator< _Ite, _Seq, _Cat > std::__fill_n_a (const __gnu_debug::Safe_iterator< _Ite,`
`_Seq, _Cat > & __first, _Size __n, const _Tp & __value, std::input_iterator_tag)`
- `template<typename _II1, typename _II2, typename _Seq2, typename _Cat2 >`
`bool std::__lexicographical_compare_aux (_II1 __first1, _II1 __last1, const __gnu_debug::Safe_iterator<`
`_II2, _Seq2, _Cat2 > & __first2, const __gnu_debug::Safe_iterator< _II2, _Seq2, _Cat2 > & __last2)`
- `template<typename _Ite1, typename _Seq1, typename _Cat1, typename _II2 >`
`bool std::__lexicographical_compare_aux (const __gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __first1,`
`const __gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __last1, _II2 __first2, _II2 __last2)`
- `template<typename _Ite1, typename _Seq1, typename _Cat1, typename _Ite2, typename _Seq2, typename _Cat2 >`
`bool std::__lexicographical_compare_aux (const __gnu_debug::Safe_iterator< _Ite1, _Seq1, ↵`
`_Cat1 > & __first1, const __gnu_debug::Safe_iterator< _Ite1, _Seq1, _Cat1 > & __last1, const`
`__gnu_debug::Safe_iterator< _Ite2, _Seq2, _Cat2 > & __first2, const __gnu_debug::Safe_iterator< ↵`
`_Ite2, _Seq2, _Cat2 > & __last2)`
- `template<typename _Ite, typename _Seq >`
`_Ite std::__niter_base (const __gnu_debug::Safe_iterator< _Ite, _Seq, std::random_access_iterator_tag >`
`&)`

7.254.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.255 `safe_local_iterator.h` File Reference

Classes

- class [__gnu_debug:: Safe_local_iterator](#)< [_Iterator](#), [_Sequence](#) >

Namespaces

- namespace [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_VERIFY_OPERANDS(_Lhs, _Rhs)`

Functions

- `template<typename _Iterator, typename _Sequence >
_Iterator __gnu_debug:: unsafe (const _Safe_local_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator, typename _Sequence >
bool __gnu_debug:: valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator< _Iterator, _Sequence > &)`
- `template<typename _Iterator, typename _Sequence >
bool __gnu_debug:: valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &, const _Safe_local_iterator< _Iterator, _Sequence > &, typename _Distance_traits< _Iterator >::__type &)`

7.255.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.256 `safe_local_iterator.tcc` File Reference

Namespaces

- namespace [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_SAFE_LOCAL_ITERATOR_TCC`

7.256.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.257 `safe_sequence.h` File Reference

Classes

- class [__gnu_debug:: After_nth_from](#)< [_Iterator](#) >
- class [__gnu_debug:: Equal_to](#)< [_Type](#) >
- class [__gnu_debug:: Not_equal_to](#)< [_Type](#) >
- class [__gnu_debug:: Safe_node_sequence](#)< [_Sequence](#) >
- class [__gnu_debug:: Safe_sequence](#)< [_Sequence](#) >

Namespaces

- namespace [__gnu_debug](#)

7.257.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.258 `safe_sequence.tcc` File Reference

Namespaces

- namespace `__gnu_debug`

Macros

- `#define _GLIBCXX_DEBUG_SAFE_SEQUENCE_TCC`

7.258.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.259 `safe_unordered_base.h` File Reference

Classes

- class `__gnu_debug::_Safe_local_iterator_base`
- class `__gnu_debug::_Safe_unordered_container_base`

Namespaces

- namespace `__gnu_debug`

7.259.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.260 `safe_unordered_container.h` File Reference

Classes

- class `__gnu_debug::_Safe_unordered_container<_Container>`

Namespaces

- namespace `__gnu_debug`

7.260.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.261 `safe_unordered_container.tcc` File Reference

Namespaces

- namespace `__gnu_debug`

Macros

- `#define _GLIBCXX_DEBUG_SAFE_UNORDERED_CONTAINER_TCC`

7.261.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.262 set.h File Reference

Classes

- class `std::__debug::set<_Key, _Compare, _Allocator>`

Namespaces

- namespace `std`
- namespace `std::__debug`

Functions

- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator!= (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator< (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator<= (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator== (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator> (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator>
bool std::__debug::operator>= (const set<_Key, _Compare, _Allocator> &__lhs, const set<_Key, _Compare, _Allocator> &__rhs)`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>
std::__debug::set (_InputIterator, _InputIterator, _Allocator) -> set< typename iterator_traits<_InputIterator>::value_type, less< typename iterator_traits<_InputIterator>::value_type>, _Allocator>`
- `template<typename _InputIterator, typename _Compare = less< typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>
std::__debug::set (_InputIterator, _InputIterator, _Compare=_Compare(), _Allocator=_Allocator()) -> set< typename iterator_traits<_InputIterator>::value_type, _Compare, _Allocator>`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>
std::__debug::set (initializer_list<_Key>, _Allocator) -> set<_Key, less<_Key>, _Allocator>`
- `template<typename _Key, typename _Compare = less<_Key>, typename _Allocator = allocator<_Key>, typename = _RequireNotAllocator<_Compare>, typename = _RequireAllocator<_Allocator>>
std::__debug::set (initializer_list<_Key>, _Compare=_Compare(), _Allocator=_Allocator()) -> set<_Key, _Compare, _Allocator>`
- `template<typename _Key, typename _Compare, typename _Allocator>
void std::__debug::swap (set<_Key, _Compare, _Allocator> &__x, set<_Key, _Compare, _Allocator> &__y)
noexcept(/*conditional */)`

7.262.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.263 decimal File Reference

Classes

- class [std::decimal::decimal128](#)
- class [std::decimal::decimal32](#)
- class [std::decimal::decimal64](#)

Namespaces

- namespace [std](#)
- namespace [std::decimal](#)

Macros

- `#define _DECLARE_DECIMAL128_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL32_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL64_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL_BINARY_OP_WITH_DEC(_Op, _T1, _T2, _T3)`
- `#define _DECLARE_DECIMAL_BINARY_OP_WITH_INT(_Op, _Tp)`
- `#define _DECLARE_DECIMAL_COMPARISON(_Op, _Tp)`
- `#define _GLIBCXX_DECIMAL`
- `#define _GLIBCXX_USE_DECIMAL_`

Functions

- double [std::decimal::decimal128_to_double](#) ([decimal128](#) __d)
- float [std::decimal::decimal128_to_float](#) ([decimal128](#) __d)
- long double [std::decimal::decimal128_to_long_double](#) ([decimal128](#) __d)
- long long [std::decimal::decimal128_to_long_long](#) ([decimal128](#) __d)
- double [std::decimal::decimal32_to_double](#) ([decimal32](#) __d)
- float [std::decimal::decimal32_to_float](#) ([decimal32](#) __d)
- long double [std::decimal::decimal32_to_long_double](#) ([decimal32](#) __d)
- long long [std::decimal::decimal32_to_long_long](#) ([decimal32](#) __d)
- double [std::decimal::decimal64_to_double](#) ([decimal64](#) __d)
- float [std::decimal::decimal64_to_float](#) ([decimal64](#) __d)
- long double [std::decimal::decimal64_to_long_double](#) ([decimal64](#) __d)
- long long [std::decimal::decimal64_to_long_long](#) ([decimal64](#) __d)
- double [std::decimal::decimal_to_double](#) ([decimal128](#) __d)
- double [std::decimal::decimal_to_double](#) ([decimal32](#) __d)
- double [std::decimal::decimal_to_double](#) ([decimal64](#) __d)
- float [std::decimal::decimal_to_float](#) ([decimal128](#) __d)
- float [std::decimal::decimal_to_float](#) ([decimal32](#) __d)
- float [std::decimal::decimal_to_float](#) ([decimal64](#) __d)
- long double [std::decimal::decimal_to_long_double](#) ([decimal128](#) __d)
- long double [std::decimal::decimal_to_long_double](#) ([decimal32](#) __d)
- long double [std::decimal::decimal_to_long_double](#) ([decimal64](#) __d)
- long long [std::decimal::decimal_to_long_long](#) ([decimal128](#) __d)
- long long [std::decimal::decimal_to_long_long](#) ([decimal32](#) __d)

- long long **std::decimal::decimal_to_long_long** (decimal64 __d)
- static decimal128 **std::decimal::make_decimal128** (long long __coeff, int __exp)
- static decimal128 **std::decimal::make_decimal128** (unsigned long long __coeff, int __exp)
- static decimal32 **std::decimal::make_decimal32** (long long __coeff, int __exp)
- static decimal32 **std::decimal::make_decimal32** (unsigned long long __coeff, int __exp)
- static decimal64 **std::decimal::make_decimal64** (long long __coeff, int __exp)
- static decimal64 **std::decimal::make_decimal64** (unsigned long long __coeff, int __exp)
- bool **std::decimal::operator!=** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator!=** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator!=** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator!=** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator!=** (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator!=** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator!=** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator!=** (unsigned long long __lhs, decimal32 __rhs)

- Generated by Doxygen

- Generated by Doxygen

- Generated by Doxygen

- `decimal64 std::decimal::operator/ (decimal32 __lhs, decimal64 __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, int __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, long __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, long long __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, unsigned int __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, unsigned long __rhs)`
- `decimal32 std::decimal::operator/ (decimal32 __lhs, unsigned long long __rhs)`
- `decimal128 std::decimal::operator/ (decimal64 __lhs, decimal128 __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, decimal64 __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, int __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, long __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, long long __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, unsigned int __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, unsigned long __rhs)`
- `decimal64 std::decimal::operator/ (decimal64 __lhs, unsigned long long __rhs)`
- `decimal128 std::decimal::operator/ (int __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (int __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (int __lhs, decimal64 __rhs)`
- `decimal128 std::decimal::operator/ (long __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (long __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (long __lhs, decimal64 __rhs)`
- `decimal128 std::decimal::operator/ (long long __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (long long __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (long long __lhs, decimal64 __rhs)`
- `decimal128 std::decimal::operator/ (unsigned int __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (unsigned int __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (unsigned int __lhs, decimal64 __rhs)`
- `decimal128 std::decimal::operator/ (unsigned long __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (unsigned long __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (unsigned long __lhs, decimal64 __rhs)`
- `decimal128 std::decimal::operator/ (unsigned long long __lhs, decimal128 __rhs)`
- `decimal32 std::decimal::operator/ (unsigned long long __lhs, decimal32 __rhs)`
- `decimal64 std::decimal::operator/ (unsigned long long __lhs, decimal64 __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, decimal128 __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, decimal32 __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, decimal64 __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, int __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, long __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, long long __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, unsigned int __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, unsigned long __rhs)`
- `bool std::decimal::operator< (decimal128 __lhs, unsigned long long __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, decimal128 __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, decimal32 __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, decimal64 __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, int __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, long __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, long long __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, unsigned int __rhs)`
- `bool std::decimal::operator< (decimal32 __lhs, unsigned long __rhs)`

- bool **std::decimal::operator<** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, long __rhs)

- [illegible]

- bool **std::decimal::operator**> (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator**> (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**> (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator**> (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator**> (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**> (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator**> (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator**> (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator**> (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator**> (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator**> (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**> (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator**> (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator**> (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**> (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator**> (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator**>= (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator**>= (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator**>= (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator**>= (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator**>= (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator**>= (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator**>= (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator**>= (long long __lhs, decimal32 __rhs)

- bool **std::decimal::operator>=** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal64 __rhs)

7.263.1 Detailed Description

This is a Standard C++ Library header.

7.264 deque File Reference

Classes

- class **std::__debug::deque**< _Tp, _Allocator >

Namespaces

- namespace **std**
- namespace **std::__debug**

Macros

- #define **_GLIBCXX_DEBUG_DEQUE**

Functions

- template<typename _InputIterator , typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>
std::__debug::deque (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> **deque**< _ValT, _Allocator >
- template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>
std::__debug::deque (size_t, _Tp, _Allocator=_Allocator()) -> **deque**< _Tp, _Allocator >
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator!=** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator<** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator<=** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator==** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator>** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **std::__debug::operator>=** (const **deque**< _Tp, _Alloc > &__lhs, const **deque**< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
void **std::__debug::swap** (**deque**< _Tp, _Alloc > &__lhs, **deque**< _Tp, _Alloc > &__rhs) noexcept(*/*conditional */*)

7.264.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.265 deque File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_erase_if`
- `#define _GLIBCXX_DEQUE`

Typedefs

- `template<typename _Tp >`
`using std::pmr::deque = std::deque< _Tp, polymorphic_allocator< _Tp > >`

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
`deque< _Tp, _Alloc >::size_type std::erase (deque< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`deque< _Tp, _Alloc >::size_type std::erase_if (deque< _Tp, _Alloc > &__cont, _Predicate __pred)`

7.265.1 Detailed Description

This is a Standard C++ Library header.

7.266 deque File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_DEQUE`

Typedefs

- `template<typename _Tp >`
`using std::experimental::fundamentals_v2::pmr::deque = std::deque< _Tp, polymorphic_allocator< _Tp >`
`>`

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
`void std::experimental::erase (deque< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void std::experimental::erase_if (deque< _Tp, _Alloc > &__cont, _Predicate __pred)`

7.266.1 Detailed Description

This is a TS C++ Library header.

7.267 expected File Reference

7.267.1 Detailed Description

This is a Standard C++ Library header.

7.268 lfts_config.h File Reference

7.268.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

7.269 numeric_traits.h File Reference

Namespaces

- namespace [__gnu_cxx](#)

Macros

- `#define __glibcxx_digits10(_Tp)`
- `#define __glibcxx_floating(_Tp, _Fval, _Dval, _LDval)`
- `#define __glibcxx_max_digits10(_Tp)`
- `#define __glibcxx_max_exponent10(_Tp)`
- `#define _GLIBCXX_INT_N_TRAITS(T, WIDTH)`

Typedefs

- `template<typename _Tp >`
`using __gnu_cxx::__int_traits = __numeric_traits_integer< _Tp >`

7.269.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.270 propagate_const File Reference

Classes

- class [std::experimental::fundamentals_v2::propagate_const< _Tp >](#)

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_PROPAGATE_CONST`

Functions

- `template<typename _Tp >`
`constexpr const _Tp & std::experimental::get_underlying (const propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Tp >`
`constexpr _Tp & std::experimental::get_underlying (propagate_const< _Tp > &__pt) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator!= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator!= (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator!= (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator< (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator< (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator< (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator<= (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator== (const propagate_const< _Tp > &__pt, nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::operator== (nullptr_t, const propagate_const< _Tp > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const propagate_const< _Tp > &__pt, const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator> (const propagate_const< _Tp > &__pt, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const _Tp &__t, const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const propagate_const< _Tp > &__pt, const _Up &__u)`

- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::operator>= (const propagate_const<_Tp> &__pt, const propagate_const<_Up> &__pu)`
- `template<typename _Tp >`
`constexpr enable_if_t< __is_swappable< _Tp >::value, void > std::experimental::swap (propagate_const<_Tp> &__pt, propagate_const<_Tp> &__pt2) noexcept(__is_nothrow_swappable< _Tp >::value)`

7.270.1 Detailed Description

This is a TS C++ Library header.

7.271 `simd` File Reference

Macros

- `#define __cpp_lib_experimental_parallel_simd`
- `#define _GLIBCXX_EXPERIMENTAL_SIMD`

7.271.1 Detailed Description

This is a TS C++ Library header.

7.272 `aligned_buffer.h` File Reference

Namespaces

- namespace [__gnu_cxx](#)

7.272.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.273 `atomicity.h` File Reference

Namespaces

- namespace [__gnu_cxx](#)

Macros

- `#define _GLIBCXX_READ_MEM_BARRIER`
- `#define _GLIBCXX_WRITE_MEM_BARRIER`

Functions

- `void __gnu_cxx::__atomic_add (volatile _Atomic_word *, int) noexcept`
- `void __gnu_cxx::__atomic_add_dispatch (_Atomic_word * __mem, int __val)`
- `void __gnu_cxx::__atomic_add_single (_Atomic_word * __mem, int __val)`
- `_Atomic_word __gnu_cxx::__exchange_and_add (volatile _Atomic_word *, int) noexcept`
- `_Atomic_word __gnu_cxx::__exchange_and_add_dispatch (_Atomic_word * __mem, int __val)`
- `_Atomic_word __gnu_cxx::__exchange_and_add_single (_Atomic_word * __mem, int __val)`
- `bool __gnu_cxx::__is_single_threaded () noexcept`

7.273.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.274 bitmap_allocator.h File Reference

Classes

- class [__gnu_cxx::__detail::__mini_vector<_Tp>](#)
- class [__gnu_cxx::__detail::__Bitmap_counter<_Tp>](#)
- class [__gnu_cxx::__detail::__Ffit_finder<_Tp>](#)
- class [__gnu_cxx::bitmap_allocator<_Tp>](#)
- class [__gnu_cxx::free_list](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [__gnu_cxx::__detail](#)

Macros

- [#define _BALLOC_ALIGN_BYTES](#)

Enumerations

- enum { [bits_per_byte](#) , [bits_per_block](#) }

Functions

- void [__gnu_cxx::__detail::__bit_allocate](#) (std::size_t *__pmap, std::size_t __pos) throw ()
- void [__gnu_cxx::__detail::__bit_free](#) (std::size_t *__pmap, std::size_t __pos) throw ()
- template<typename _ForwardIterator, typename _Tp, typename _Compare >
_ForwardIterator [__gnu_cxx::__detail::__lower_bound](#) (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)
- template<typename _AddrPair >
std::size_t [__gnu_cxx::__detail::__num_bitmaps](#) (_AddrPair __ap)
- template<typename _AddrPair >
std::size_t [__gnu_cxx::__detail::__num_blocks](#) (_AddrPair __ap)
- std::size_t [__gnu_cxx::__Bit_scan_forward](#) (std::size_t __num)
- template<typename _Tp1, typename _Tp2 >
bool [__gnu_cxx::operator==](#) (const [bitmap_allocator](#)<_Tp1 > &, const [bitmap_allocator](#)<_Tp2 > &) throw ()

7.274.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.274.2 Macro Definition Documentation

[_BALLOC_ALIGN_BYTES](#)

```
#define _BALLOC_ALIGN_BYTES
```

The constant in the expression below is the alignment required in bytes.

7.275 cast.h File Reference

Classes

- struct [__gnu_cxx::_Caster<_ToType>](#)

Namespaces

- namespace [__gnu_cxx](#)

Functions

- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__const_pointer_cast (_FromType *__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__const_pointer_cast (const _FromType &__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__dynamic_pointer_cast (_FromType *__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__dynamic_pointer_cast (const _FromType &__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__reinterpret_pointer_cast (_FromType *__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__reinterpret_pointer_cast (const _FromType &__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__static_pointer_cast (_FromType *__arg)`
- `template<typename _ToType, typename _FromType >
_ToType __gnu_cxx::__static_pointer_cast (const _FromType &__arg)`

7.275.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/pointer.h>`.

7.276 `codecvt_specializations.h` File Reference

Classes

- class `std::codecvt< _InternT, _ExternT, encoding_state >`
- struct `__gnu_cxx::encoding_char_traits< _CharT >`
- class `__gnu_cxx::encoding_state`

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Functions

- `template<typename _Tp >
size_t std::__iconv_adaptor (size_t(*__func)(iconv_t, _Tp, size_t *, char **, size_t *), iconv_t __cd, char **↔
__inbuf, size_t *__inbytes, char **__outbuf, size_t *__outbytes)`

7.276.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.277 `concurrency.h` File Reference

Classes

- class `__gnu_cxx::__scoped_lock`

Namespaces

- namespace [__gnu_cxx](#)

Enumerations

- enum `_Lock_policy` { `_S_single` , `_S_mutex` , `_S_atomic` }

Functions

- void `__gnu_cxx::__throw_concurrency_lock_error()`
- void `__gnu_cxx::__throw_concurrency_unlock_error()`

Variables

- const `_Lock_policy` `__gnu_cxx::__default_lock_policy`

7.277.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.278 debug_allocator.h File Reference

Classes

- class [__gnu_cxx::debug_allocator<_Alloc>](#)

Namespaces

- namespace [__gnu_cxx](#)

7.278.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.279 enc_filebuf.h File Reference

Classes

- class [__gnu_cxx::enc_filebuf<_CharT>](#)

Namespaces

- namespace [__gnu_cxx](#)

7.279.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.280 extptr_allocator.h File Reference

Classes

- class [__gnu_cxx::ExtPtr_allocator<_Tp>](#)

Namespaces

- namespace `__gnu_cxx`

Functions

- `template<typename _Tp >`
`void __gnu_cxx::swap (_ExtPtr_allocator< _Tp > &__larg, _ExtPtr_allocator< _Tp > &__rarg)`

7.280.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Author

Bob Walters

An example allocator which uses an alternative pointer type from `bits/pointer.h`. Supports test cases which confirm container support for alternative pointers.

7.281 malloc_allocator.h File Reference

Classes

- class `__gnu_cxx::malloc_allocator< _Tp >`

Namespaces

- namespace `__gnu_cxx`

7.281.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.282 mt_allocator.h File Reference

Classes

- `struct __gnu_cxx::__common_pool_policy< _PoolTp, _Thread >`
- `class __gnu_cxx::__mt_alloc< _Tp, _Poolp >`
- `class __gnu_cxx::__mt_alloc_base< _Tp >`
- `struct __gnu_cxx::__per_type_pool_policy< _Tp, _PoolTp, _Thread >`
- `class __gnu_cxx::__pool< false >`
- `class __gnu_cxx::__pool< true >`
- `struct __gnu_cxx::__pool_base`

Namespaces

- namespace `__gnu_cxx`

Macros

- `#define __thread_default`

Typedefs

- `typedef void(* __gnu_cxx::__destroy_handler) (void *)`

Functions

- `template<typename _Tp, typename _Poolp >`
`bool __gnu_cxx::operator== (const __mt_alloc< _Tp, _Poolp > &, const __mt_alloc< _Tp, _Poolp > &)`

7.282.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.283 `assoc_container.hpp` File Reference

Classes

- class `__gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >`
- class `__gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >`
- class `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >`
- class `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >`
- class `__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >`
- class `__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >`
- class `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >`

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_BRANCH_BASE`
- `#define PB_DS_CC_HASH_BASE`
- `#define PB_DS_GP_HASH_BASE`
- `#define PB_DS_HASH_BASE`
- `#define PB_DS_LU_BASE`
- `#define PB_DS_TREE_BASE`
- `#define PB_DS_TREE_NODE_AND_IT_TRAITS`
- `#define PB_DS_TRIE_BASE`
- `#define PB_DS_TRIE_NODE_AND_IT_TRAITS`

7.283.1 Detailed Description

Contains associative containers.

7.284 `bin_search_tree.hpp` File Reference

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node)`
- `#define PB_DS_BIN_TREE_NAME`
- `#define PB_DS_BIN_TREE_TRAITS_BASE`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_STRUCT_ONLY_ASSERT_VALID(X)`

7.284.1 Detailed Description

Contains an implementation class for binary search tree.

7.285 node_iterators.hpp File Reference

Classes

- class [__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >](#)
- class [__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_TREE_CONST_NODE_ITERATOR_CLASS_C_DEC`
- `#define PB_DS_TREE_NODE_ITERATOR_CLASS_C_DEC`

7.285.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.286 node_iterators.hpp File Reference

Classes

- class [__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >](#)
- class [__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_OV_TREE_CONST_NODE_ITERATOR_C_DEC`
- `#define PB_DS_OV_TREE_NODE_ITERATOR_C_DEC`

7.286.1 Detailed Description

Contains an implementation class for `ov_tree_`.

7.287 point_iterators.hpp File Reference

Classes

- class [__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, _Alloc >](#)
- class [__gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_TREE_CONST_IT_C_DEC`
- `#define PB_DS_TREE_CONST_ODIR_IT_C_DEC`
- `#define PB_DS_TREE_IT_C_DEC`
- `#define PB_DS_TREE_ODIR_IT_C_DEC`

7.287.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.288 r_erase_fn_imps.hpp File Reference

7.288.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.289 r_erase_fn_imps.hpp File Reference

7.289.1 Detailed Description

Contains an implementation class for `pat_trie`.

7.290 rotate_fn_imps.hpp File Reference

7.290.1 Detailed Description

Contains imps for rotating nodes.

7.291 rotate_fn_imps.hpp File Reference

7.291.1 Detailed Description

Contains imps for rotating nodes.

7.292 traits.hpp File Reference

Classes

- `struct __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >`
- `struct __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >`

Namespaces

- namespace `__gnu_pbds`

7.292.1 Detailed Description

Contains an implementation for `bin_search_tree_`.

7.293 traits.hpp File Reference

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_DEBUG_VERIFY(_Cond)`

7.293.1 Detailed Description

Contains an implementation class for tree-like classes.

7.294 traits.hpp File Reference**Classes**

- [struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`](#)
- [struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`](#)

Namespaces

- namespace [__gnu_pbds](#)

7.294.1 Detailed Description

Contains an implementation class for `ov_tree_`.

7.295 traits.hpp File Reference**Classes**

- [struct `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >`](#)
- [struct `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >`](#)

Namespaces

- namespace [__gnu_pbds](#)

7.295.1 Detailed Description

Contains an implementation class for `pat_trie_`.

7.296 traits.hpp File Reference**Classes**

- [struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`](#)
- [struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`](#)

Namespaces

- namespace [__gnu_pbds](#)

7.296.1 Detailed Description

Contains an implementation for `rb_tree_`.

7.297 traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >](#)
- struct [__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.297.1 Detailed Description

Contains an implementation for splay_tree_.

7.298 binary_heap_.hpp File Reference

Classes

- class [__gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`
- `#define PB_DS_ENTRY_CMP_DEC`
- `#define PB_DS_RESIZE_POLICY_DEC`

7.298.1 Detailed Description

Contains an implementation class for a binary heap.

7.299 const_iterator.hpp File Reference

Classes

- class [__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_BIN_HEAP_CIT_BASE`

7.299.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

7.300 `const_iterator.hpp` File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_BASIC_HEAP_CIT_BASE`
- `#define PB_DS_CLASS_C_DEC`

7.300.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

7.301 `const_iterator.hpp` File Reference

7.301.1 Detailed Description

Contains an iterator class used for const ranging over the elements of the table.

This file is intended to be included inside a class definition, with `PB_DS_CLASS_C_DEC` defined to the name of the enclosing class.

7.302 `constructors_destructor_fn_imps.hpp` File Reference

7.302.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.303 `constructors_destructor_fn_imps.hpp` File Reference

7.303.1 Detailed Description

Contains an implementation class for `binary_heap_`.

7.304 `constructors_destructor_fn_imps.hpp` File Reference

7.304.1 Detailed Description

Contains an implementation for `binomial_heap_`.

7.305 `constructors_destructor_fn_imps.hpp` File Reference

7.305.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.306 `constructors_destructor_fn_imps.hpp` File Reference

7.306.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

7.307 constructors_destructor_fn_imps.hpp File Reference

7.307.1 Detailed Description

Contains an implementation class for ov_tree_.

7.308 constructors_destructor_fn_imps.hpp File Reference

7.308.1 Detailed Description

Contains an implementation class for a pairing heap.

7.309 constructors_destructor_fn_imps.hpp File Reference

7.309.1 Detailed Description

Contains an implementation class for pat_trie.

7.310 constructors_destructor_fn_imps.hpp File Reference

7.310.1 Detailed Description

Contains an implementation for rb_tree_.

7.311 constructors_destructor_fn_imps.hpp File Reference

7.311.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

7.312 constructors_destructor_fn_imps.hpp File Reference

7.312.1 Detailed Description

Contains an implementation class for splay_tree_.

7.313 constructors_destructor_fn_imps.hpp File Reference

7.313.1 Detailed Description

Contains an implementation for thin_heap_.

7.314 debug_fn_imps.hpp File Reference

7.314.1 Detailed Description

Contains an implementation class for bin_search_tree_.

7.315 debug_fn_imps.hpp File Reference

7.315.1 Detailed Description

Contains an implementation class for a binary_heap.

7.316 debug_fn_imps.hpp File Reference

7.316.1 Detailed Description

Contains an implementation for binomial_heap_.

7.317 debug_fn_imps.hpp File Reference

7.317.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.318 debug_fn_imps.hpp File Reference

7.318.1 Detailed Description

Contains implementations of cc_ht_map_'s debug-mode functions.

7.319 debug_fn_imps.hpp File Reference

7.319.1 Detailed Description

Contains implementations of gp_ht_map_'s debug-mode functions.

7.320 debug_fn_imps.hpp File Reference

7.320.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

7.321 debug_fn_imps.hpp File Reference

7.321.1 Detailed Description

Contains implementations of cc_ht_map_'s debug-mode functions.

7.322 debug_fn_imps.hpp File Reference

7.322.1 Detailed Description

Contains an implementation class for ov_tree_.

7.323 debug_fn_imps.hpp File Reference

7.323.1 Detailed Description

Contains an implementation class for a pairing heap.

7.324 debug_fn_imps.hpp File Reference

7.324.1 Detailed Description

Contains an implementation class for pat_trie_.

7.325 debug_fn_imps.hpp File Reference

7.325.1 Detailed Description

Contains an implementation for rb_tree_.

7.326 debug_fn_imps.hpp File Reference

7.326.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

7.327 `debug_fn_imps.hpp` File Reference

7.327.1 Detailed Description

Contains an implementation class for `splay_tree_`.

7.328 `debug_fn_imps.hpp` File Reference

7.328.1 Detailed Description

Contains an implementation for `thin_heap_`.

7.329 `entry_cmp.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false>](#)
- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true>](#)
- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false>::type](#)

Namespaces

- namespace [__gnu_pbds](#)

7.329.1 Detailed Description

Contains an implementation class for a `binary_heap`.

7.330 `entry_pred.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false>](#)
- struct [__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, true>](#)

Namespaces

- namespace [__gnu_pbds](#)

7.330.1 Detailed Description

Contains an implementation class for a `binary_heap`.

7.331 `erase_fn_imps.hpp` File Reference

7.331.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.332 `erase_fn_imps.hpp` File Reference

7.332.1 Detailed Description

Contains an implementation class for a `binary_heap`.

7.333 erase_fn_imps.hpp File Reference

7.333.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.334 erase_fn_imps.hpp File Reference

7.334.1 Detailed Description

Contains implementations of cc_ht_map_'s erase related functions.

7.335 erase_fn_imps.hpp File Reference

7.335.1 Detailed Description

Contains implementations of gp_ht_map_'s erase related functions.

7.336 erase_fn_imps.hpp File Reference

7.336.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

7.337 erase_fn_imps.hpp File Reference

7.337.1 Detailed Description

Contains implementations of lu_map_.

7.338 erase_fn_imps.hpp File Reference

7.338.1 Detailed Description

Contains an implementation class for ov_tree_.

7.339 erase_fn_imps.hpp File Reference

7.339.1 Detailed Description

Contains an implementation class for a pairing heap.

7.340 erase_fn_imps.hpp File Reference

7.340.1 Detailed Description

Contains an implementation class for pat_trie.

7.341 erase_fn_imps.hpp File Reference

7.341.1 Detailed Description

Contains an implementation for rb_tree_.

7.342 erase_fn_imps.hpp File Reference

7.342.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

7.343 erase_fn_imps.hpp File Reference

7.343.1 Detailed Description

Contains an implementation class for splay_tree_.

7.344 erase_fn_imps.hpp File Reference

7.344.1 Detailed Description

Contains an implementation for thin_heap_.

7.345 find_fn_imps.hpp File Reference

7.345.1 Detailed Description

Contains an implementation class for bin_search_tree_.

7.346 find_fn_imps.hpp File Reference

7.346.1 Detailed Description

Contains an implementation class for a binary_heap.

7.347 find_fn_imps.hpp File Reference

7.347.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.348 find_fn_imps.hpp File Reference

7.348.1 Detailed Description

Contains implementations of cc_ht_map_'s find related functions.

7.349 find_fn_imps.hpp File Reference

7.349.1 Detailed Description

Contains implementations of gp_ht_map_'s find related functions.

7.350 find_fn_imps.hpp File Reference

7.350.1 Detailed Description

Contains implementations of lu_map_.

7.351 find_fn_imps.hpp File Reference

7.351.1 Detailed Description

Contains an implementation class for a pairing heap.

7.352 find_fn_imps.hpp File Reference

7.352.1 Detailed Description

Contains an implementation class for pat_trie.

7.353 find_fn_imps.hpp File Reference**7.353.1 Detailed Description**

Contains an implementation for rb_tree_.

7.354 find_fn_imps.hpp File Reference**7.354.1 Detailed Description**

Contains an implementation class for splay_tree_.

7.355 find_fn_imps.hpp File Reference**7.355.1 Detailed Description**

Contains an implementation for thin_heap_.

7.356 info_fn_imps.hpp File Reference**7.356.1 Detailed Description**

Contains an implementation class for bin_search_tree_.

7.357 info_fn_imps.hpp File Reference**7.357.1 Detailed Description**

Contains an implementation class for a binary_heap.

7.358 info_fn_imps.hpp File Reference**7.358.1 Detailed Description**

Contains implementations of cc_ht_map_'s entire container info related functions.

7.359 info_fn_imps.hpp File Reference**7.359.1 Detailed Description**

Contains implementations of gp_ht_map_'s entire container info related functions.

7.360 info_fn_imps.hpp File Reference**7.360.1 Detailed Description**

Contains an implementation class for left_child_next_sibling_heap_.

7.361 info_fn_imps.hpp File Reference**7.361.1 Detailed Description**

Contains implementations of lu_map_.

7.362 info_fn_imps.hpp File Reference**7.362.1 Detailed Description**

Contains an implementation class for ov_tree_.

7.363 info_fn_imps.hpp File Reference

7.363.1 Detailed Description

Contains an implementation class for pat_trie.

7.364 info_fn_imps.hpp File Reference

7.364.1 Detailed Description

Contains an implementation for rb_tree_.

7.365 info_fn_imps.hpp File Reference

7.365.1 Detailed Description

Contains an implementation.

7.366 insert_fn_imps.hpp File Reference

7.366.1 Detailed Description

Contains an implementation class for bin_search_tree_.

7.367 insert_fn_imps.hpp File Reference

7.367.1 Detailed Description

Contains an implementation class for a binary_heap.

7.368 insert_fn_imps.hpp File Reference

7.368.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.369 insert_fn_imps.hpp File Reference

7.369.1 Detailed Description

Contains implementations of cc_ht_map_'s insert related functions.

7.370 insert_fn_imps.hpp File Reference

7.370.1 Detailed Description

Contains implementations of gp_ht_map_'s insert related functions.

7.371 insert_fn_imps.hpp File Reference

7.371.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

7.372 insert_fn_imps.hpp File Reference

7.372.1 Detailed Description

Contains implementations of lu_map_.

7.373 insert_fn_imps.hpp File Reference

7.373.1 Detailed Description

Contains an implementation class for `ov_tree_`.

7.374 insert_fn_imps.hpp File Reference

7.374.1 Detailed Description

Contains an implementation class for a pairing heap.

7.375 insert_fn_imps.hpp File Reference

7.375.1 Detailed Description

Contains an implementation for `rb_tree_`.

7.376 insert_fn_imps.hpp File Reference

7.376.1 Detailed Description

Contains an implementation for `rc_binomial_heap_`.

7.377 insert_fn_imps.hpp File Reference

7.377.1 Detailed Description

Contains an implementation class for `splay_tree_`.

7.378 insert_fn_imps.hpp File Reference

7.378.1 Detailed Description

Contains an implementation for `thin_heap_`.

7.379 iterators_fn_imps.hpp File Reference

7.379.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

7.380 iterators_fn_imps.hpp File Reference

7.380.1 Detailed Description

Contains an implementation class for a `binary_heap`.

7.381 iterators_fn_imps.hpp File Reference

7.381.1 Detailed Description

Contains implementations of `cc_ht_map_`'s iterators related functions, e.g., `begin()`.

7.382 iterators_fn_imps.hpp File Reference

7.382.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

7.383 iterators_fn_imps.hpp File Reference

7.383.1 Detailed Description

Contains implementations of lu_map_.

7.384 iterators_fn_imps.hpp File Reference

7.384.1 Detailed Description

Contains an implementation class for ov_tree_.

7.385 iterators_fn_imps.hpp File Reference

7.385.1 Detailed Description

Contains an implementation class for pat_trie.

7.386 point_const_iterator.hpp File Reference

Classes

- class [__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.386.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

7.387 point_const_iterator.hpp File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.387.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

7.388 point_const_iterator.hpp File Reference

7.388.1 Detailed Description

Contains an iterator class returned by the tables' const find and insert methods.

- This file is intended to be included inside a class definition, with PB_DS_CLASS_C_DEC defined to the name of the enclosing class.

7.389 policy_access_fn_imps.hpp File Reference

7.389.1 Detailed Description

Contains an implementation class for bin_search_tree_.

7.390 policy_access_fn_imps.hpp File Reference

7.390.1 Detailed Description

Contains an implementation class for a binary_heap.

7.391 policy_access_fn_imps.hpp File Reference

7.391.1 Detailed Description

Contains implementations of cc_ht_map_'s policy access functions.

7.392 policy_access_fn_imps.hpp File Reference

7.392.1 Detailed Description

Contains implementations of gp_ht_map_'s policy access functions.

7.393 policy_access_fn_imps.hpp File Reference

7.393.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

7.394 policy_access_fn_imps.hpp File Reference

7.394.1 Detailed Description

Contains an implementation class for ov_tree.

7.395 policy_access_fn_imps.hpp File Reference

7.395.1 Detailed Description

Contains an implementation class for pat_trie.

7.396 resize_policy.hpp File Reference

Classes

- class [__gnu_pbds::detail::resize_policy< _Tp >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.396.1 Detailed Description

Contains an implementation class for a binary_heap.

7.397 split_join_fn_imps.hpp File Reference

7.397.1 Detailed Description

Contains an implementation class for bin_search_tree_.

7.398 split_join_fn_imps.hpp File Reference

7.398.1 Detailed Description

Contains an implementation class for a binary_heap.

7.399 split_join_fn_imps.hpp File Reference

7.399.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.400 split_join_fn_imps.hpp File Reference

7.400.1 Detailed Description

Contains an implementation class for ov_tree_.

7.401 split_join_fn_imps.hpp File Reference

7.401.1 Detailed Description

Contains an implementation class for a pairing heap.

7.402 split_join_fn_imps.hpp File Reference

7.402.1 Detailed Description

Contains an implementation for rb_tree_.

7.403 split_join_fn_imps.hpp File Reference

7.403.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

7.404 split_join_fn_imps.hpp File Reference

7.404.1 Detailed Description

Contains an implementation class for splay_tree_.

7.405 split_join_fn_imps.hpp File Reference

7.405.1 Detailed Description

Contains an implementation for thin_heap_.

7.406 trace_fn_imps.hpp File Reference

7.406.1 Detailed Description

Contains an implementation class for a binary_heap.

7.407 `trace_fn_imps.hpp` File Reference

7.407.1 Detailed Description

Contains implementations of `cc_ht_map_`'s trace-mode functions.

7.408 `trace_fn_imps.hpp` File Reference

7.408.1 Detailed Description

Contains implementations of `gp_ht_map_`'s trace-mode functions.

7.409 `trace_fn_imps.hpp` File Reference

7.409.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

7.410 `trace_fn_imps.hpp` File Reference

7.410.1 Detailed Description

Contains implementations of `lu_map_`.

7.411 `trace_fn_imps.hpp` File Reference

7.411.1 Detailed Description

Contains an implementation class for `pat_trie_`.

7.412 `trace_fn_imps.hpp` File Reference

7.412.1 Detailed Description

Contains an implementation for `rc_binomial_heap_`.

7.413 `trace_fn_imps.hpp` File Reference

7.413.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

7.414 `binomial_heap_.hpp` File Reference

Classes

- class [`__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >`](#)

Namespaces

- namespace [`__gnu_pbds`](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.414.1 Detailed Description

Contains an implementation class for a binomial heap.

7.415 binomial_heap_base.hpp File Reference

Classes

- class [__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- #define **PB_DS_ASSERT_BASE_NODE_CONSISTENT**(_Node, _Bool)
- #define **PB_DS_ASSERT_VALID_COND**(X, _StrictlyBinomial)
- #define **PB_DS_B_HEAP_BASE**
- #define **PB_DS_CLASS_C_DEC**
- #define **PB_DS_CLASS_T_DEC**

7.415.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

7.416 branch_policy.hpp File Reference

Classes

- struct [__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >](#)
- struct [__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.416.1 Detailed Description

Contains a base class for branch policies.

7.417 null_node_metadata.hpp File Reference

Classes

- struct [__gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.417.1 Detailed Description

Contains an implementation class for tree-like classes.

7.418 cc_ht_map_.hpp File Reference

Classes

- class [__gnu_pbds::detail::cc_ht_map](#)< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >

Namespaces

- namespace [__gnu_pbds](#)

Macros

- #define **PB_DS_CC_HASH_NAME**
- #define **PB_DS_CC_HASH_TRAITS_BASE**
- #define **PB_DS_CLASS_C_DEC**
- #define **PB_DS_CLASS_T_DEC**
- #define **PB_DS_GEN_POS**
- #define **PB_DS_HASH_EQ_FN_C_DEC**
- #define **PB_DS_RANGED_HASH_FN_C_DEC**

7.418.1 Detailed Description

Contains an implementation class for cc_ht_map_.

7.419 cmp_fn_imps.hpp File Reference

7.419.1 Detailed Description

Contains implementations of cc_ht_map_'s entire container comparison related functions.

7.420 cond_key_dtor_entry_dealtor.hpp File Reference

Classes

- class [__gnu_pbds::detail::cond_dealtor](#)< Entry, _Alloc >

Namespaces

- namespace [__gnu_pbds](#)

7.420.1 Detailed Description

Contains a conditional key destructor, used for exception handling.

7.421 constructor_destructor_fn_imps.hpp File Reference

7.421.1 Detailed Description

Contains implementations of cc_ht_map_'s constructors, destructor, and related functions.

7.422 constructor_destructor_fn_imps.hpp File Reference

7.422.1 Detailed Description

Contains implementations of gp_ht_map_'s constructors, destructor, and related functions.

7.423 constructor_destructor_fn_imps.hpp File Reference**7.424 constructor_destructor_no_store_hash_fn_imps.hpp File Reference****7.424.1 Detailed Description**

Contains implementations of cc_ht_map_'s constructors, destructor, and related functions.

7.425 constructor_destructor_no_store_hash_fn_imps.hpp File Reference**7.425.1 Detailed Description**

Contains implementations of gp_ht_map_'s constructors, destructor, and related functions.

7.426 constructor_destructor_store_hash_fn_imps.hpp File Reference**7.426.1 Detailed Description**

Contains implementations of cc_ht_map_'s constructors, destructor, and related functions.

7.427 constructor_destructor_store_hash_fn_imps.hpp File Reference**7.427.1 Detailed Description**

Contains implementations of gp_ht_map_'s constructors, destructor, and related functions.

7.428 debug_no_store_hash_fn_imps.hpp File Reference**7.428.1 Detailed Description**

Contains implementations of cc_ht_map_'s debug-mode functions.

7.429 debug_no_store_hash_fn_imps.hpp File Reference**7.429.1 Detailed Description**

Contains implementations of gp_ht_map_'s debug-mode functions.

7.430 debug_store_hash_fn_imps.hpp File Reference**7.430.1 Detailed Description**

Contains implementations of cc_ht_map_'s debug-mode functions.

7.431 debug_store_hash_fn_imps.hpp File Reference**7.431.1 Detailed Description**

Contains implementations of gp_ht_map_'s debug-mode functions.

7.432 entry_list_fn_imps.hpp File Reference**7.432.1 Detailed Description**

Contains implementations of cc_ht_map_'s entry-list related functions.

7.433 erase_no_store_hash_fn_imps.hpp File Reference**7.433.1 Detailed Description**

Contains implementations of `cc_ht_map_'s` erase related functions, when the hash value is not stored.

7.434 erase_no_store_hash_fn_imps.hpp File Reference**7.434.1 Detailed Description**

Contains implementations of `gp_ht_map_'s` erase related functions, when the hash value is not stored.

7.435 erase_store_hash_fn_imps.hpp File Reference**7.435.1 Detailed Description**

Contains implementations of `cc_ht_map_'s` erase related functions, when the hash value is stored.

7.436 erase_store_hash_fn_imps.hpp File Reference**7.436.1 Detailed Description**

Contains implementations of `gp_ht_map_'s` erase related functions, when the hash value is stored.

7.437 find_store_hash_fn_imps.hpp File Reference**7.437.1 Detailed Description**

Contains implementations of `cc_ht_map_'s` find related functions, when the hash value is stored.

7.438 find_store_hash_fn_imps.hpp File Reference**7.438.1 Detailed Description**

Contains implementations of `gp_ht_map_'s` insert related functions, when the hash value is stored.

7.439 insert_no_store_hash_fn_imps.hpp File Reference**7.439.1 Detailed Description**

Contains implementations of `cc_ht_map_'s` insert related functions, when the hash value is not stored.

7.440 insert_no_store_hash_fn_imps.hpp File Reference**7.440.1 Detailed Description**

Contains implementations of `gp_ht_map_'s` insert related functions, when the hash value is not stored.

7.441 insert_store_hash_fn_imps.hpp File Reference**7.441.1 Detailed Description**

Contains implementations of `cc_ht_map_'s` insert related functions, when the hash value is stored.

7.442 insert_store_hash_fn_imps.hpp File Reference**7.442.1 Detailed Description**

Contains implementations of `gp_ht_map_'s` find related functions, when the hash value is stored.

7.443 `resize_fn_imps.hpp` File Reference

7.443.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions.

7.444 `resize_fn_imps.hpp` File Reference

7.444.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions.

7.445 `resize_no_store_hash_fn_imps.hpp` File Reference

7.445.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions, when the hash value is not stored.

7.446 `resize_no_store_hash_fn_imps.hpp` File Reference

7.446.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions, when the hash value is not stored.

7.447 `resize_store_hash_fn_imps.hpp` File Reference

7.447.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions, when the hash value is stored.

7.448 `resize_store_hash_fn_imps.hpp` File Reference

7.448.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions, when the hash value is stored.

7.449 `size_fn_imps.hpp` File Reference

7.449.1 Detailed Description

Contains implementations of `cc_ht_map_`'s entire container size related functions.

7.450 `cond_dealtor.hpp` File Reference

Classes

- class [`__gnu_pbds::detail::cond_dealtor< Entry, _Alloc >`](#)

Namespaces

- namespace [`__gnu_pbds`](#)

7.450.1 Detailed Description

Contains a conditional deallocator.

7.451 container_base_dispatch.hpp File Reference

Classes

- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_TI >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_CHECK_KEY_DOES_NOT_EXIST(_Key)`
- `#define PB_DS_CHECK_KEY_EXISTS(_Key)`
- `#define PB_DS_DATA_FALSE_INDICATOR`
- `#define PB_DS_DATA_TRUE_INDICATOR`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`
- `#define PB_DS_EP2VP(X)`
- `#define PB_DS_EP2VP(X)`
- `#define PB_DS_V2F(X)`
- `#define PB_DS_V2F(X)`
- `#define PB_DS_V2S(X)`
- `#define PB_DS_V2S(X)`

7.451.1 Detailed Description

Contains associative container dispatching.

7.452 debug_map_base.hpp File Reference

7.452.1 Detailed Description

Contains a debug-mode base for all maps.

7.453 eq_by_less.hpp File Reference

Classes

- struct [__gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.453.1 Detailed Description

Contains an equivalence function.

7.454 hash_eq_fn.hpp File Reference

Classes

- struct [__gnu_pbds::detail::hash_eq_fn](#)< Key, Eq_Fn, _Alloc, false >
- struct [__gnu_pbds::detail::hash_eq_fn](#)< Key, Eq_Fn, _Alloc, true >

Namespaces

- namespace [__gnu_pbds](#)

7.454.1 Detailed Description

Contains 2 equivalence functions, one employing a hash value, and one ignoring it.

7.455 find_no_store_hash_fn_imps.hpp File Reference

7.455.1 Detailed Description

Contains implementations of gp_ht_map_'s find related functions, when the hash value is not stored.

7.456 gp_ht_map_.hpp File Reference

Classes

- class [__gnu_pbds::detail::gp_ht_map](#)< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_GEN_POS`
- `#define PB_DS_GP_HASH_NAME`
- `#define PB_DS_GP_HASH_TRAITS_BASE`
- `#define PB_DS_HASH_EQ_FN_C_DEC`
- `#define PB_DS_RANGED_PROBE_FN_C_DEC`

7.456.1 Detailed Description

Contains an implementation class for general probing hash.

7.457 `iterator_fn_imps.hpp` File Reference

7.457.1 Detailed Description

Contains implementations of `gp_ht_map_'s` iterators related functions, e.g., `begin()`.

7.458 `direct_mask_range_hashing_imp.hpp` File Reference

7.458.1 Detailed Description

Contains a range-hashing policy implementation

7.459 `direct_mod_range_hashing_imp.hpp` File Reference

7.459.1 Detailed Description

Contains a range-hashing policy implementation

7.460 `linear_probe_fn_imp.hpp` File Reference

7.460.1 Detailed Description

Contains a probe policy implementation

7.461 `mask_based_range_hashing.hpp` File Reference

Classes

- class [__gnu_pbds::detail::mask_based_range_hashing< Size_Type >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.461.1 Detailed Description

Contains a range hashing policy base.

7.462 `mod_based_range_hashing.hpp` File Reference

Classes

- class [__gnu_pbds::detail::mod_based_range_hashing< Size_Type >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.462.1 Detailed Description

Contains a range hashing policy base.

7.463 `probe_fn_base.hpp` File Reference

Classes

- class [__gnu_pbds::detail::probe_fn_base< _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.463.1 Detailed Description

Contains a probe policy base.

7.464 quadratic_probe_fn_imp.hpp File Reference**7.464.1 Detailed Description**

Contains a probe policy implementation

7.465 ranged_hash_fn.hpp File Reference**Classes**

- class [__gnu_pbds::detail::ranged_hash_fn](#)< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >
- class [__gnu_pbds::detail::ranged_hash_fn](#)< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >
- class [__gnu_pbds::detail::ranged_hash_fn](#)< Key, null_type, _Alloc, Comb_Hash_Fn, false >
- class [__gnu_pbds::detail::ranged_hash_fn](#)< Key, null_type, _Alloc, Comb_Hash_Fn, true >

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.465.1 Detailed Description

Contains a unified ranged hash functor, allowing the hash tables to deal with a single class for ranged hashing.

7.466 ranged_probe_fn.hpp File Reference**Classes**

- class [__gnu_pbds::detail::ranged_probe_fn](#)< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >
- class [__gnu_pbds::detail::ranged_probe_fn](#)< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >
- class [__gnu_pbds::detail::ranged_probe_fn](#)< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.466.1 Detailed Description

Contains a unified ranged probe functor, allowing the probe tables to deal with a single class for ranged probeing.

7.467 `sample_probe_fn.hpp` File Reference**Classes**

- class [__gnu_pbds::sample_probe_fn](#)

Namespaces

- namespace [__gnu_pbds](#)

7.467.1 Detailed Description

Contains a sample probe policy.

7.468 `sample_range_hashing.hpp` File Reference**Classes**

- class [__gnu_pbds::sample_range_hashing](#)

Namespaces

- namespace [__gnu_pbds](#)

7.468.1 Detailed Description

Contains a range hashing policy.

7.469 `sample_ranged_hash_fn.hpp` File Reference**Classes**

- class [__gnu_pbds::sample_ranged_hash_fn](#)

Namespaces

- namespace [__gnu_pbds](#)

7.469.1 Detailed Description

Contains a ranged hash policy.

7.470 `sample_ranged_probe_fn.hpp` File Reference**Classes**

- class [__gnu_pbds::sample_ranged_probe_fn](#)

Namespaces

- namespace [__gnu_pbds](#)

7.470.1 Detailed Description

Contains a ranged probe policy.

7.471 left_child_next_sibling_heap_.hpp File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.471.1 Detailed Description

Contains an implementation class for a basic heap.

7.472 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::left_child_next_sibling_heap_node_< _Value, _Metadata, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.472.1 Detailed Description

Contains an implementation struct for this type of heap's node.

7.473 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.473.1 Detailed Description

Contains an implementation for rb_tree_.

7.474 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::splay_tree_node_< Value_Type, Metadata, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.474.1 Detailed Description

Contains an implementation struct for splay_tree_'s node.

7.475 entry_metadata_base.hpp File Reference

Namespaces

- namespace [__gnu_pbds](#)

7.475.1 Detailed Description

Contains an implementation for a list update map.

7.476 lu_map_.hpp File Reference

Classes

- class [__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_GEN_POS`
- `#define PB_DS_LU_NAME`
- `#define PB_DS_LU_TRAITS_BASE`

7.476.1 Detailed Description

Contains a list update map.

7.477 lu_counter_metadata.hpp File Reference

Classes

- class [__gnu_pbds::detail::lu_counter_metadata< Size_Type >](#)
- class [__gnu_pbds::detail::lu_counter_policy_base< Size_Type >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.477.1 Detailed Description

Contains implementation of a lu counter policy's metadata.

7.478 sample_update_policy.hpp File Reference

Classes

- struct [__gnu_pbds::sample_update_policy](#)

Namespaces

- namespace [__gnu_pbds](#)

7.478.1 Detailed Description

Contains a sample policy for list update containers.

7.479 ov_tree_map_.hpp File Reference

Classes

- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >](#)
- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CONST_NODE_ITERATOR_NAME`
- `#define PB_DS_OV_TREE_NAME`
- `#define PB_DS_OV_TREE_TRAITS_BASE`

7.479.1 Detailed Description

Contains an implementation class for ov_tree.

7.480 pairing_heap_.hpp File Reference

Classes

- class [__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node, _Bool)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_P_HEAP_BASE`

7.480.1 Detailed Description

Contains an implementation class for a pairing heap.

7.481 `insert_join_fn_imps.hpp` File Reference

7.481.1 Detailed Description

Contains an implementation class for `pat_trie`.

7.482 `pat_trie.hpp` File Reference

Classes

- class `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >`

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_ASSERT_NODE_VALID(X)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_PAT_TRIE_NAME`
- `#define PB_DS_PAT_TRIE_TRAITS_BASE`
- `#define PB_DS_RECURSIVE_COUNT_LEAFS(X)`

7.482.1 Detailed Description

Contains an implementation class for a patricia tree.

7.483 `pat_trie_base.hpp` File Reference

Classes

- class `__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`
- struct `__gnu_pbds::detail::pat_trie_base::_Head< _ATraits, Metadata >`
- struct `__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >`
- class `__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`
- struct `__gnu_pbds::detail::pat_trie_base::_Leaf< _ATraits, Metadata >`
- struct `__gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >`
- struct `__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >`
- struct `__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >`
- class `__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >`
- class `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >`
- struct `__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator`
- struct `__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::iterator`
- struct `__gnu_pbds::detail::pat_trie_base`

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CONST_IT_C_DEC`
- `#define PB_DS_CONST_ODIR_IT_C_DEC`
- `#define PB_DS_IT_C_DEC`
- `#define PB_DS_ODIR_IT_C_DEC`
- `#define PB_DS_PAT_TRIE_NODE_CONST_ITERATOR_C_DEC`
- `#define PB_DS_PAT_TRIE_NODE_ITERATOR_C_DEC`

7.483.1 Detailed Description

Contains the base class for a patricia tree.

7.484 split_fn_imps.hpp File Reference**7.484.1 Detailed Description**

Contains an implementation class for pat_trie.

7.485 synth_access_traits.hpp File Reference**Classes**

- struct [__gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_SYNTH_E_ACCESS_TRAITS_C_DEC`
- `#define PB_DS_SYNTH_E_ACCESS_TRAITS_T_DEC`

7.485.1 Detailed Description

Contains an implementation class for a patricia tree.

7.486 update_fn_imps.hpp File Reference**7.486.1 Detailed Description**

Contains an implementation class for pat_trie_.

7.487 priority_queue_base_dispatch.hpp File Reference**Classes**

- struct [__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type >](#)
- struct [__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`

7.487.1 Detailed Description

Contains an pqiative container dispatching base.

7.488 `rb_tree_.hpp` File Reference

Classes

- class [__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_RB_TREE_BASE`
- `#define PB_DS_RB_TREE_BASE_NAME`
- `#define PB_DS_RB_TREE_NAME`
- `#define PB_DS_STRUCT_ONLY_ASSERT_VALID(X)`

7.488.1 Detailed Description

Contains an implementation for Red Black trees.

7.489 `rc.hpp` File Reference

Classes

- class [__gnu_pbds::detail::rc< _Node, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.489.1 Detailed Description

Contains a redundant (binary counter).

7.490 `rc_binomial_heap_.hpp` File Reference

Classes

- class [__gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_RC_C_DEC`

7.490.1 Detailed Description

Contains an implementation for redundant-counter binomial heap.

7.491 cc_hash_max_collision_check_resize_trigger_imp.hpp File Reference

7.491.1 Detailed Description

Contains a resize trigger implementation.

7.492 hash_exponential_size_policy_imp.hpp File Reference

7.492.1 Detailed Description

Contains a resize size policy implementation.

7.493 hash_load_check_resize_trigger_imp.hpp File Reference

7.493.1 Detailed Description

Contains a resize trigger implementation.

7.494 hash_load_check_resize_trigger_size_base.hpp File Reference

Classes

- class [__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.494.1 Detailed Description

Contains an base holding size for some resize policies.

7.495 hash_prime_size_policy_imp.hpp File Reference

7.495.1 Detailed Description

Contains a resize size policy implementation.

7.496 hash_standard_resize_policy_imp.hpp File Reference

7.496.1 Detailed Description

Contains a resize policy implementation.

7.497 `sample_resize_policy.hpp` File Reference

Classes

- class [__gnu_pbds::sample_resize_policy](#)

Namespaces

- namespace [__gnu_pbds](#)

7.497.1 Detailed Description

Contains a sample resize policy for hash tables.

7.498 `sample_resize_trigger.hpp` File Reference

Classes

- class [__gnu_pbds::sample_resize_trigger](#)

Namespaces

- namespace [__gnu_pbds](#)

7.498.1 Detailed Description

Contains a sample resize trigger policy class.

7.499 `sample_size_policy.hpp` File Reference

Classes

- class [__gnu_pbds::sample_size_policy](#)

Namespaces

- namespace [__gnu_pbds](#)

7.499.1 Detailed Description

Contains a sample size resize-policy.

7.500 `splay_fn_imps.hpp` File Reference

7.500.1 Detailed Description

Contains an implementation class for `splay_tree_`.

7.501 `splay_tree_.hpp` File Reference

Classes

- class [__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_BASE_NODE_CONSISTENT(_Node)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_S_TREE_BASE`
- `#define PB_DS_S_TREE_BASE_NAME`
- `#define PB_DS_S_TREE_NAME`

7.501.1 Detailed Description

Contains an implementation class for splay trees.

7.502 standard_policies.hpp File Reference**Classes**

- struct [__gnu_pbds::detail::default_comb_hash_fn](#)
- struct [__gnu_pbds::detail::default_eq_fn< Key >](#)
- struct [__gnu_pbds::detail::default_hash_fn< Key >](#)
- struct [__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >](#)
- struct [__gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >](#)
- struct [__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >](#)
- struct [__gnu_pbds::detail::default_update_policy](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define __dtrie_alloc`
- `#define __dtrie_string`

Enumerations

- enum { **default_store_hash** }

7.502.1 Detailed Description

Contains standard policies for containers.

7.502.2 Enumeration Type Documentation**anonymous enum**

`anonymous enum`

Enumeration for default behavior of stored hash data.

7.503 thin_heap.hpp File Reference**Classes**

- class [__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_AUX_NULL(X)`
- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node, _Bool)`
- `#define PB_DS_BASE_T_P`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

Enumerations

- enum { `num_distinct_rank_bounds` }

Variables

- static const std::size_t `__gnu_pbds::detail::g_a_rank_bounds` [num_distinct_rank_bounds]

7.503.1 Detailed Description

Contains an implementation class for a thin heap.

7.504 node_metadata_selector.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >](#)
- struct [__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >](#)
- struct [__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.504.1 Detailed Description

Contains an implementation class for trees.

7.505 node_metadata_selector.hpp File Reference

Classes

- struct [__gnu_pbds::detail::trie_metadata_helper< Node_Update, false >](#)
- struct [__gnu_pbds::detail::trie_metadata_helper< Node_Update, true >](#)
- struct [__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.505.1 Detailed Description

Contains an implementation class for tries.

7.506 order_statistics_imp.hpp File Reference

7.506.1 Detailed Description

Contains forward declarations for order_statistics_key

7.507 order_statistics_imp.hpp File Reference

7.507.1 Detailed Description

Contains forward declarations for order_statistics_key

7.508 sample_tree_node_update.hpp File Reference

Classes

- class [__gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.508.1 Detailed Description

Contains a samle node update functor.

7.509 tree_trace_base.hpp File Reference

7.509.1 Detailed Description

Contains tree-related policies.

7.510 prefix_search_node_update_imp.hpp File Reference

7.510.1 Detailed Description

Contains an implementation of prefix_search_node_update.

7.511 sample_trie_access_traits.hpp File Reference

Classes

- struct [__gnu_pbds::sample_trie_access_traits](#)

Namespaces

- namespace [__gnu_pbds](#)

7.511.1 Detailed Description

Contains a sample probe policy.

7.512 sample_trie_node_update.hpp File Reference

Classes

- class [__gnu_pbds::sample_trie_node_update< Node_CItr, Node_Itr, ATraits, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.512.1 Detailed Description

Contains a samle node update functor.

7.513 trie_policy_base.hpp File Reference**Classes**

- class [__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

7.513.1 Detailed Description

Contains an implementation of `trie_policy_base`.

7.514 trie_string_access_traits_imp.hpp File Reference**7.514.1 Detailed Description**

Contains a policy for extracting character positions from a string for a vector-based PATRICIA tree

7.515 type_utils.hpp File Reference**Namespaces**

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_STATIC_ASSERT(UNIQUE, E)`

Typedefs

- `typedef std::tr1::integral_constant< int, 0 > __gnu_pbds::detail::false_type`
- `typedef std::tr1::integral_constant< int, 1 > __gnu_pbds::detail::true_type`

7.515.1 Detailed Description

Contains utilities for handling types. All of these classes are based on Modern C++ by Andrei Alexandrescu.

7.516 types_traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash >](#)
- struct [__gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash >](#)
- struct [__gnu_pbds::detail::no_throw_copies< Key, Mapped >](#)
- struct [__gnu_pbds::detail::no_throw_copies< Key, null_type >](#)
- struct [__gnu_pbds::detail::rebind_traits< _Alloc, T >](#)
- struct [__gnu_pbds::detail::select_value_type< Key, Mapped >](#)
- struct [__gnu_pbds::detail::select_value_type< Key, null_type >](#)
- struct [__gnu_pbds::detail::stored_data< _Tv, _Th, Store_Hash >](#)
- struct [__gnu_pbds::detail::stored_data< _Tv, _Th, false >](#)
- struct [__gnu_pbds::detail::stored_hash< _Th >](#)
- struct [__gnu_pbds::detail::stored_value< _Tv >](#)
- struct [__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.516.1 Detailed Description

Contains a traits class of types used by containers.

7.517 iterator.hpp File Reference

7.517.1 Detailed Description

Contains an iterator_class used for ranging over the elements of the table.

This file is intended to be included inside a class definition, with PB_DS_CLASS_C_DEC defined to the name of the enclosing class.

7.518 point_iterator.hpp File Reference

7.518.1 Detailed Description

Contains an iterator class returned by the tables' find and insert methods.

This file is intended to be included inside a class definition, with PB_DS_CLASS_C_DEC defined to the name of the enclosing class.

7.519 exception.hpp File Reference

Classes

- struct [__gnu_pbds::container_error](#)
- struct [__gnu_pbds::insert_error](#)
- struct [__gnu_pbds::join_error](#)
- struct [__gnu_pbds::resize_error](#)

Namespaces

- namespace [__gnu_pbds](#)

Functions

- void `__gnu_pbds::__throw_container_error()`
- void `__gnu_pbds::__throw_insert_error()`
- void `__gnu_pbds::__throw_join_error()`
- void `__gnu_pbds::__throw_resize_error()`

7.519.1 Detailed Description

Contains exception classes.

7.520 hash_policy.hpp File Reference

Classes

- class `__gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >`
- class `__gnu_pbds::direct_mask_range_hashing< Size_Type >`
- class `__gnu_pbds::direct_mod_range_hashing< Size_Type >`
- class `__gnu_pbds::hash_exponential_size_policy< Size_Type >`
- class `__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >`
- class `__gnu_pbds::hash_prime_size_policy`
- class `__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >`
- class `__gnu_pbds::linear_probe_fn< Size_Type >`
- class `__gnu_pbds::quadratic_probe_fn< Size_Type >`

Namespaces

- namespace `__gnu_pbds`

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_SIZE_BASE_C_DEC`

7.520.1 Detailed Description

Contains hash-related policies.

7.521 list_update_policy.hpp File Reference

Classes

- class [__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >](#)
- class [__gnu_pbds::lu_move_to_front_policy< _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.521.1 Detailed Description

Contains policies for list update containers.

7.522 priority_queue.hpp File Reference

Classes

- class [__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

7.522.1 Detailed Description

Contains priority_queues.

7.523 tag_and_trait.hpp File Reference

Classes

- struct [__gnu_pbds::associative_tag](#)
- struct [__gnu_pbds::basic_branch_tag](#)
- struct [__gnu_pbds::basic_hash_tag](#)
- struct [__gnu_pbds::basic_invalidation_guarantee](#)
- struct [__gnu_pbds::binary_heap_tag](#)
- struct [__gnu_pbds::binomial_heap_tag](#)
- struct [__gnu_pbds::cc_hash_tag](#)
- struct [__gnu_pbds::container_tag](#)
- struct [__gnu_pbds::container_traits< Cntnr >](#)
- struct [__gnu_pbds::container_traits_base< binary_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< binomial_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< cc_hash_tag >](#)
- struct [__gnu_pbds::container_traits_base< gp_hash_tag >](#)
- struct [__gnu_pbds::container_traits_base< list_update_tag >](#)
- struct [__gnu_pbds::container_traits_base< ov_tree_tag >](#)
- struct [__gnu_pbds::container_traits_base< pairing_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< pat_trie_tag >](#)
- struct [__gnu_pbds::container_traits_base< rb_tree_tag >](#)
- struct [__gnu_pbds::container_traits_base< rc_binomial_heap_tag >](#)
- struct [__gnu_pbds::container_traits_base< splay_tree_tag >](#)
- struct [__gnu_pbds::container_traits_base< thin_heap_tag >](#)
- struct [__gnu_pbds::gp_hash_tag](#)

- struct [__gnu_pbds::list_update_tag](#)
- struct [__gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 >](#)
- struct [__gnu_pbds::null_type](#)
- struct [__gnu_pbds::ov_tree_tag](#)
- struct [__gnu_pbds::pairing_heap_tag](#)
- struct [__gnu_pbds::pat_trie_tag](#)
- struct [__gnu_pbds::point_invalidation_guarantee](#)
- struct [__gnu_pbds::priority_queue_tag](#)
- struct [__gnu_pbds::range_invalidation_guarantee](#)
- struct [__gnu_pbds::rb_tree_tag](#)
- struct [__gnu_pbds::rc_binomial_heap_tag](#)
- struct [__gnu_pbds::sequence_tag](#)
- struct [__gnu_pbds::splay_tree_tag](#)
- struct [__gnu_pbds::string_tag](#)
- struct [__gnu_pbds::thin_heap_tag](#)
- struct [__gnu_pbds::tree_tag](#)
- struct [__gnu_pbds::trie_tag](#)
- struct [__gnu_pbds::trivial_iterator_tag](#)

Namespaces

- namespace [__gnu_pbds](#)

Typedefs

- typedef void [__gnu_pbds::trivial_iterator_difference_type](#)

7.523.1 Detailed Description

Contains tags and traits, e.g., ones describing underlying data structures.

7.524 tree_policy.hpp File Reference

Classes

- class [__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >](#)

Namespaces

- namespace [__gnu_pbds](#)

Macros

- #define **PB_DS_BRANCH_POLICY_BASE**
- #define **PB_DS_CLASS_C_DEC**
- #define **PB_DS_CLASS_T_DEC**

7.524.1 Detailed Description

Contains tree-related policies.

7.525 trie_policy.hpp File Reference

Classes

- class [__gnu_pbds::trie_order_statistics_node_update](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >
- class [__gnu_pbds::trie_prefix_search_node_update](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >
- struct [__gnu_pbds::trie_string_access_traits](#)< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >

Namespaces

- namespace [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_TRIE_POLICY_BASE`

7.525.1 Detailed Description

Contains trie-related policies.

7.526 pod_char_traits.h File Reference

Classes

- struct [std::char_traits](#)< [__gnu_cxx::character](#)< _Value, _Int, _St > >
- struct [__gnu_cxx::character](#)< _Value, _Int, _St >

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Functions

- `template<typename _Value , typename _Int , typename _St >`
`bool __gnu_cxx::operator< (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St >`
`&rhs)`
- `template<typename _Value , typename _Int , typename _St >`
`bool __gnu_cxx::operator== (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St >`
`&rhs)`

7.526.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.527 pointer.h File Reference

Classes

- struct [__gnu_cxx::Invalid_type](#)
- class [__gnu_cxx::Pointer_adapter<_Storage_policy>](#)
- class [__gnu_cxx::Relative_pointer_impl<_Tp>](#)
- class [__gnu_cxx::Relative_pointer_impl<const _Tp>](#)
- class [__gnu_cxx::Std_pointer_impl<_Tp>](#)
- struct [__gnu_cxx::Unqualified_type<_Tp>](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Macros

- `#define CXX_POINTER_ARITH_OPERATOR_SET(INT_TYPE)`
- `#define GCC_CXX_POINTER_COMPARISON_OPERATION_SET(OPERATOR)`

Functions

- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator!= (_Tp1 __lhs, const Pointer_adapter<_Tp2> &__rhs)`
- `template<typename _Tp>`
`bool __gnu_cxx::operator!= (const Pointer_adapter<_Tp> &__lhs, const Pointer_adapter<_Tp> &__rhs)`
- `template<typename _Tp>`
`bool __gnu_cxx::operator!= (const Pointer_adapter<_Tp> &__lhs, int __rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator!= (const Pointer_adapter<_Tp1> &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator!= (const Pointer_adapter<_Tp1> &__lhs, const Pointer_adapter<_Tp2> &__rhs)`
- `template<typename _Tp>`
`bool __gnu_cxx::operator!= (int __lhs, const Pointer_adapter<_Tp> &__rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator< (_Tp1 __lhs, const Pointer_adapter<_Tp2> &__rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator< (const Pointer_adapter<_Tp1> &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator< (const Pointer_adapter<_Tp1> &__lhs, const Pointer_adapter<_Tp2> &__rhs)`
- `template<typename _CharT, typename _Traits, typename _StoreT>`
`std::basic_ostream<_CharT, _Traits> &__gnu_cxx::operator<< (std::basic_ostream<_CharT, _Traits> &__os, const Pointer_adapter<_StoreT> &p)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator<= (_Tp1 __lhs, const Pointer_adapter<_Tp2> &__rhs)`
- `template<typename _Tp>`
`bool __gnu_cxx::operator<= (const Pointer_adapter<_Tp> &__lhs, const Pointer_adapter<_Tp> &__rhs)`
- `template<typename _Tp1, typename _Tp2>`
`bool __gnu_cxx::operator<= (const Pointer_adapter<_Tp1> &__lhs, _Tp2 __rhs)`

- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator<= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`

7.527.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Author

Bob Walters

Provides reusable `_Pointer_adapter` for assisting in the development of custom pointer types that can be used with the standard containers via the `allocator::pointer` and `allocator::const_pointer` typedefs.

7.528 pool_allocator.h File Reference

Classes

- class `__gnu_cxx::__pool_alloc< _Tp >`
- class `__gnu_cxx::__pool_alloc_base`

Namespaces

- namespace [__gnu_cxx](#)

Functions

- `template<typename _Tp >
bool __gnu_cxx::operator== (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`

7.528.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.529 [rb_tree](#) File Reference

Classes

- struct [__gnu_cxx::rb_tree](#)< [_Key](#), [_Value](#), [_KeyOfValue](#), [_Compare](#), [_Alloc](#) >

Namespaces

- namespace [__gnu_cxx](#)

Macros

- `#define _RB_TREE`

7.529.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.530 [rc_string_base.h](#) File Reference

Classes

- class [__gnu_cxx::__rc_string_base](#)< [_CharT](#), [_Traits](#), [_Alloc](#) >

Namespaces

- namespace [__gnu_cxx](#)

7.530.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

7.531 [rope](#) File Reference

Classes

- class [__gnu_cxx::rope](#)< [_CharT](#), [_Alloc](#) >

Namespaces

- namespace [__gnu_cxx](#)
- namespace [__gnu_cxx::__detail](#)
- namespace [std](#)
- namespace [std::tr1](#)

Macros

- #define **__GC_CONST**
- #define **__ROPE_DEFINE_ALLOC**(_Tp, __name)
- #define **__ROPE_DEFINE_ALLOC**(_Tp, __name)
- #define **__ROPE_DEFINE_ALLOCS**(__a)
- #define **__STATIC_IF_SGI_ALLOC**
- #define **__STL_FREE_STRING**(__s, __l, __a)
- #define **__STL_ROPE_FROM_UNOWNED_CHAR_PTR**(__s, __size, __a)
- #define **__ROPE**

Typedefs

- typedef [rope](#)< char > **__gnu_cxx::crope**
- typedef [rope](#)< wchar_t > **__gnu_cxx::wrope**

Enumerations

- enum { **_S_max_rope_depth** }
- enum **_Tag** { **_S_leaf** , **_S_concat** , **_S_substringfn** , **_S_function** }

Functions

- crope::reference **__gnu_cxx::__mutable_reference_at** ([crope](#) &__c, std::size_t __i)
- template<typename _ForwardIterator, typename _Allocator >
void **__gnu_cxx::__Destroy_const** (_ForwardIterator __first, _ForwardIterator __last, _Allocator __alloc)
- template<typename _ForwardIterator, typename _Tp >
void **__gnu_cxx::__Destroy_const** (_ForwardIterator __first, _ForwardIterator __last, [std::allocator](#)< _Tp >)
- template<class _CharT >
void **__gnu_cxx::__S_cond_store_eos** (_CharT &)
- void **__gnu_cxx::__S_cond_store_eos** (char &__c)
- void **__gnu_cxx::__S_cond_store_eos** (wchar_t &__c)
- template<class _CharT >
_CharT **__gnu_cxx::__S_eos** (_CharT *)
- template<class _CharT >
bool **__gnu_cxx::__S_is_basic_char_type** (_CharT *)
- bool **__gnu_cxx::__S_is_basic_char_type** (char *)
- bool **__gnu_cxx::__S_is_basic_char_type** (wchar_t *)
- template<class _CharT >
bool **__gnu_cxx::__S_is_one_byte_char_type** (_CharT *)
- bool **__gnu_cxx::__S_is_one_byte_char_type** (char *)
- template<class _CharT, class _Alloc >
bool **__gnu_cxx::operator!=** (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)
- template<class _CharT, class _Alloc >
bool **__gnu_cxx::operator!=** (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)
- template<class _CharT, class _Alloc >
bool **__gnu_cxx::operator!=** (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)
- template<class _CharT, class _Alloc >
bool **__gnu_cxx::operator!=** (const [rope](#)< _CharT, _Alloc > &__x, const [rope](#)< _CharT, _Alloc > &__y)

- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (const _Rope_const_iterator< _CharT, _Alloc`
`> &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (const _Rope_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, const _CharT *__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (std::ptrdiff_t __n, const _Rope_const_iterator< _CharT, _Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (std::ptrdiff_t __n, const _Rope_iterator< _CharT, _Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, const _CharT *__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`std::ptrdiff_t __gnu_cxx::operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`std::ptrdiff_t __gnu_cxx::operator- (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator- (const _Rope_iterator< _CharT, _Alloc > &__x, std::ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__o, const rope< _CharT, _Alloc > &__r)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`

- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`void __gnu_cxx::swap (_Rope_char_ref_proxy< _CharT, _Alloc > __a, _Rope_char_ref_proxy< _CharT, _Alloc > __b)`
- `template<class _CharT, class _Alloc >`
`void __gnu_cxx::swap (rope< _CharT, _Alloc > &__x, rope< _CharT, _Alloc > &__y)`

Variables

- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::identity_element (_Rope_Concat_fn< _CharT, _Alloc >)`

7.531.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.532 ropeimpl.h File Reference

Namespaces

- namespace `__gnu_cxx`

Functions

- `template<class _CharT, class _Traits >`
`void __gnu_cxx::Rope_fill (std::basic_ostream< _CharT, _Traits > &__o, std::size_t __n)`
- `template<class _CharT >`
`bool __gnu_cxx::Rope_is_simple (_CharT *)`
- `bool __gnu_cxx::Rope_is_simple (char *)`
- `bool __gnu_cxx::Rope_is_simple (wchar_t *)`
- `template<class _Rope_iterator >`
`void __gnu_cxx::Rope_rotate (_Rope_iterator __first, _Rope_iterator __middle, _Rope_iterator __last)`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > &__gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__o, const rope< _CharT, _Alloc > &__r)`
- `void __gnu_cxx::rotate (_Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __first, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __middle, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __last)`

7.532.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/rope>`.

7.533 slist File Reference

Classes

- class `__gnu_cxx::slist< _Tp, _Alloc >`

Namespaces

- namespace `__gnu_cxx`
- namespace `std`

Macros

- `#define _SLIST`

Functions

- `_Slist_node_base * __gnu_cxx::__slist_make_link (_Slist_node_base * __prev_node, _Slist_node_base * __new_node)`
- `_Slist_node_base * __gnu_cxx::__slist_previous (_Slist_node_base * __head, const _Slist_node_base * __node)`
- `const _Slist_node_base * __gnu_cxx::__slist_previous (const _Slist_node_base * __head, const _Slist_node_base * __node)`
- `_Slist_node_base * __gnu_cxx::__slist_reverse (_Slist_node_base * __node)`
- `std::size_t __gnu_cxx::__slist_size (_Slist_node_base * __node)`
- `void __gnu_cxx::__slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __before_first, _Slist_node_base * __before_last)`
- `void __gnu_cxx::__slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __head)`
- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator!= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator< (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`

- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator<= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator== (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator> (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc >`
`bool __gnu_cxx::operator>= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc >`
`void __gnu_cxx::swap (slist< _Tp, _Alloc > &__x, slist< _Tp, _Alloc > &__y)`

7.533.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.534 sso_string_base.h File Reference

Namespaces

- namespace `__gnu_cxx`

7.534.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

7.535 stdio_filebuf.h File Reference

Classes

- class `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`

Namespaces

- namespace `__gnu_cxx`

7.535.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.536 stdio_sync_filebuf.h File Reference

Classes

- class `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`

Namespaces

- namespace `__gnu_cxx`

7.536.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.537 `string_conversions.h` File Reference

Namespaces

- namespace [__gnu_cxx](#)

Functions

- `template<typename _TRet, typename _Ret = _TRet, typename _CharT, typename... _Base>
_Ret __gnu_cxx::__sto(_TRet(*__convf)(const _CharT *, _CharT **, _Base...), const char *__name, const
_CharT *__str, std::size_t *__idx, _Base... __base)`
- `template<typename _String, typename _CharT = typename _String::value_type>
_String __gnu_cxx::__to_xstring(int(*__convf)(_CharT *, std::size_t, const _CharT *, __builtin_va_list), std::size_t __n, const _CharT *__fmt,...)`

7.537.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.538 `throw_allocator.h` File Reference

Classes

- struct [__gnu_cxx::limit_condition::always_adjustor](#)
- struct [__gnu_cxx::random_condition::always_adjustor](#)
- struct [__gnu_cxx::annotate_base](#)
- struct [__gnu_cxx::condition_base](#)
- struct [__gnu_cxx::forced_error](#)
- struct [__gnu_cxx::random_condition::group_adjustor](#)
- struct `std::hash< __gnu_cxx::throw_value_limit >`
- struct `std::hash< __gnu_cxx::throw_value_random >`
- struct [__gnu_cxx::limit_condition::limit_adjustor](#)
- struct [__gnu_cxx::limit_condition](#)
- struct [__gnu_cxx::random_condition::never_adjustor](#)
- struct [__gnu_cxx::limit_condition::never_adjustor](#)
- struct [__gnu_cxx::random_condition](#)
- class [__gnu_cxx::throw_allocator_base](#)< _Tp, _Cond >
- struct [__gnu_cxx::throw_allocator_limit](#)< _Tp >
- struct [__gnu_cxx::throw_allocator_random](#)< _Tp >
- struct [__gnu_cxx::throw_value_base](#)< _Cond >
- struct [__gnu_cxx::throw_value_limit](#)
- struct [__gnu_cxx::throw_value_random](#)

Namespaces

- namespace [__gnu_cxx](#)
- namespace [std](#)

Functions

- `void __gnu_cxx::__throw_forced_error ()`
- `template<typename _Cond >
throw_value_base< _Cond > __gnu_cxx::operator* (const throw_value_base< _Cond > &__a, const
throw_value_base< _Cond > &__b)`

- `template<typename _Cond >`
`throw_value_base< _Cond > __gnu_cxx::operator+ (const throw_value_base< _Cond > &__a, const`
`throw_value_base< _Cond > &__b)`
- `template<typename _Cond >`
`throw_value_base< _Cond > __gnu_cxx::operator- (const throw_value_base< _Cond > &__a, const`
`throw_value_base< _Cond > &__b)`
- `template<typename _Cond >`
`bool __gnu_cxx::operator< (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond >`
`&__b)`
- `std::ostream & __gnu_cxx::operator<< (std::ostream &os, const annotate_base &__b)`
- `template<typename _Tp, typename _Cond >`
`bool __gnu_cxx::operator== (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp,`
`_Cond > &)`
- `template<typename _Cond >`
`bool __gnu_cxx::operator== (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond >`
`&__b)`
- `template<typename _Cond >`
`void __gnu_cxx::swap (throw_value_base< _Cond > &__a, throw_value_base< _Cond > &__b)`

7.538.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Contains two exception-generating types (`throw_value`, `throw_allocator`) intended to be used as value and allocator types while testing exception safety in templated containers and algorithms. The allocator has additional log and debug features. The exception generated is of type `forced_exception_error`.

7.539 `type_traits.h` File Reference

Namespaces

- namespace `__gnu_cxx`

Functions

- `template<typename _Type >`
`constexpr bool __gnu_cxx::__is_null_pointer (_Type *__ptr)`
- `template<typename _Type >`
`constexpr bool __gnu_cxx::__is_null_pointer (_Type)`
- `constexpr bool __gnu_cxx::__is_null_pointer (std::nullptr_t)`

7.539.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.540 `typelist.h` File Reference

Namespaces

- namespace `__gnu_cxx`
- namespace `__gnu_cxx::typelist`

Macros

- `#define _GLIBCXX_TYPELIST_CHAIN1(X0)`
- `#define _GLIBCXX_TYPELIST_CHAIN10(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9)`
- `#define _GLIBCXX_TYPELIST_CHAIN11(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10)`
- `#define _GLIBCXX_TYPELIST_CHAIN12(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11)`
- `#define _GLIBCXX_TYPELIST_CHAIN13(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12)`
- `#define _GLIBCXX_TYPELIST_CHAIN14(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13)`
- `#define _GLIBCXX_TYPELIST_CHAIN15(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14)`
- `#define _GLIBCXX_TYPELIST_CHAIN16(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15)`
- `#define _GLIBCXX_TYPELIST_CHAIN17(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16)`
- `#define _GLIBCXX_TYPELIST_CHAIN18(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17)`
- `#define _GLIBCXX_TYPELIST_CHAIN19(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17, X18)`
- `#define _GLIBCXX_TYPELIST_CHAIN2(X0, X1)`
- `#define _GLIBCXX_TYPELIST_CHAIN20(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17, X18, X19)`
- `#define _GLIBCXX_TYPELIST_CHAIN3(X0, X1, X2)`
- `#define _GLIBCXX_TYPELIST_CHAIN4(X0, X1, X2, X3)`
- `#define _GLIBCXX_TYPELIST_CHAIN5(X0, X1, X2, X3, X4)`
- `#define _GLIBCXX_TYPELIST_CHAIN6(X0, X1, X2, X3, X4, X5)`
- `#define _GLIBCXX_TYPELIST_CHAIN7(X0, X1, X2, X3, X4, X5, X6)`
- `#define _GLIBCXX_TYPELIST_CHAIN8(X0, X1, X2, X3, X4, X5, X6, X7)`
- `#define _GLIBCXX_TYPELIST_CHAIN9(X0, X1, X2, X3, X4, X5, X6, X7, X8)`

Functions

- `template<typename Fn, typename Typelist>`
`void __gnu_cxx::typelist::apply (Fn &, Typelist)`
- `template<typename Fn, typename Typelist>`
`void __gnu_cxx::typelist::apply_generator (Fn &fn, Typelist)`
- `template<typename Fn, typename TypelistT, typename TypelistV>`
`void __gnu_cxx::typelist::apply_generator (Fn &fn, TypelistT, TypelistV)`
- `template<typename Gn, typename Typelist>`
`void __gnu_cxx::typelist::apply_generator (Gn &, Typelist)`
- `template<typename Gn, typename TypelistT, typename TypelistV>`
`void __gnu_cxx::typelist::apply_generator (Gn &, TypelistT, TypelistV)`

7.540.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Contains `typelist_chain` definitions. Typelists are an idea by Andrei Alexandrescu.

7.541 `vstring.h` File Reference

Classes

- class `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>`
- struct `std::hash<__gnu_cxx::__u16vstring>`
- struct `std::hash<__gnu_cxx::__u32vstring>`
- struct `std::hash<__gnu_cxx::__vstring>`
- struct `std::hash<__gnu_cxx::__wvstring>`

Namespaces

- namespace `__gnu_cxx`
- namespace `std`

Functions

- [illegible]

- Generated by Doxygen

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool __gnu_cxx::operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT`
`*__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool __gnu_cxx::operator>= (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base >`
`&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa_string<`
`_CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`void __gnu_cxx::swap (__versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, __versa_string< _CharT, _`
`Traits, _Alloc, _Base > &__rhs)`

7.541.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.542 vstring.tcc File Reference

Namespaces

- namespace `__gnu_cxx`
- namespace `std`

Macros

- `#define _VSTRING_TCC`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa_string<`
`_CharT, _Traits, _Alloc, _Base > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (_CharT __lhs, const __versa_string<`
`_CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const _CharT *__lhs, const`
`__versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa_string<`
`_CharT, _Traits, _Alloc, _Base > &__str)`

7.542.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

7.543 vstring_fwd.h File Reference

Namespaces

- namespace `__gnu_cxx`

Typedefs

- typedef `__versa_string`< char, `std::char_traits`< char >, `std::allocator`< char >, `__rc_string_base` > `__gnu_cxx::__rc_string`
- typedef `__vstring` `__gnu_cxx::__sso_string`
- typedef `__versa_string`< char16_t, `std::char_traits`< char16_t >, `std::allocator`< char16_t >, `__rc_string_base` > `__gnu_cxx::__u16rc_string`
- typedef `__u16vstring` `__gnu_cxx::__u16sso_string`
- typedef `__versa_string`< char16_t > `__gnu_cxx::__u16vstring`
- typedef `__versa_string`< char32_t, `std::char_traits`< char32_t >, `std::allocator`< char32_t >, `__rc_string_base` > `__gnu_cxx::__u32rc_string`
- typedef `__u32vstring` `__gnu_cxx::__u32sso_string`
- typedef `__versa_string`< char32_t > `__gnu_cxx::__u32vstring`
- typedef `__versa_string`< char > `__gnu_cxx::__vstring`
- typedef `__versa_string`< wchar_t, `std::char_traits`< wchar_t >, `std::allocator`< wchar_t >, `__rc_string_base` > `__gnu_cxx::__wrc_string`
- typedef `__wvstring` `__gnu_cxx::__wsso_string`
- typedef `__versa_string`< wchar_t > `__gnu_cxx::__wvstring`

7.543.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

7.544 vstring_util.h File Reference

Namespaces

- namespace `__gnu_cxx`

7.544.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

7.545 fenv.h File Reference

7.545.1 Detailed Description

This is a Standard C++ Library header.

7.546 filesystem File Reference

Macros

- `#define __cpp_lib_experimental_filesystem`
- `#define _GLIBCXX_EXPERIMENTAL_FILESYSTEM`

7.546.1 Detailed Description

This is a TS C++ Library header.

7.547 filesystem File Reference

Macros

- `#define __cpp_lib_filesystem`
- `#define _GLIBCXX_FILESYSTEM`

7.547.1 Detailed Description

This is a Standard C++ Library header.

7.548 forward_list File Reference

Classes

- class `__gnu_debug::__Safe_forward_list<_SafeSequence >`
- class `std::__debug::forward_list<_Tp, _Alloc >`

Namespaces

- namespace `__gnu_debug`
- namespace `std`
- namespace `std::__debug`

Macros

- `#define __glibcxx_check_valid_fl_range(_First, _Last, _Dist)`
- `#define _GLIBCXX20_ONLY(__expr)`
- `#define _GLIBCXX_DEBUG_FORWARD_LIST`
- `#define _GLIBCXX_FWDLIST_REMOVE_RETURN_TYPE_TAG`

Functions

- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`std::__debug::forward_list (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> forward_list<_ValT, _↵`
`Allocator >`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>>`
`std::__debug::forward_list (size_t, _Tp, _Allocator=_Allocator()) -> forward_list<_Tp, _Allocator >`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator!= (const forward_list<_Tp, _Alloc > &__lx, const forward_list<_Tp, _Alloc >`
`&__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator< (const forward_list<_Tp, _Alloc > &__lx, const forward_list<_Tp, _Alloc >`
`&__ly)`

- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`void std::__debug::swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.swap(__ly)))`

7.548.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.549 forward_list File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define _GLIBCXX_EXPERIMENTAL_FORWARD_LIST`

Typedefs

- `template<typename _Tp >`
`using std::experimental::fundamentals_v2::pmr::forward_list = std::forward_list< _Tp, polymorphic_allocator< _Tp > >`

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
`void std::experimental::erase (forward_list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void std::experimental::erase_if (forward_list< _Tp, _Alloc > &__cont, _Predicate __pred)`

7.549.1 Detailed Description

This is a TS C++ Library header.

7.550 forward_list File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_erase_if`
- `#define _GLIBCXX_FORWARD_LIST`

Typedefs

- `template<typename _Tp >`
`using std::pmr::forward_list = std::forward_list< _Tp, polymorphic_allocator< _Tp > >`

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
`forward_list< _Tp, _Alloc >::size_type std::erase (forward_list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`forward_list< _Tp, _Alloc >::size_type std::erase_if (forward_list< _Tp, _Alloc > &__cont, _Predicate __pred)`

7.550.1 Detailed Description

This is a Standard C++ Library header.

7.551 fstream File Reference**Classes**

- class `std::basic_filebuf< _CharT, _Traits >`
- class `std::basic_fstream< _CharT, _Traits >`
- class `std::basic_ifstream< _CharT, _Traits >`
- class `std::basic_ofstream< _CharT, _Traits >`

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_BUFSIZ`
- `#define _GLIBCXX_FSTREAM`

Typedefs

- `template<typename _Path, typename _Result = _Path, typename _Path2 = decltype(std::declval<_Path&>().make_preferred()).filename()>`
`using std::_If_fs_path = enable_if_t< is_same_v< _Path, _Path2 >, _Result >`

Functions

- `template<class _CharT, class _Traits >`
`void std::swap (basic_filebuf< _CharT, _Traits > &__x, basic_filebuf< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_fstream< _CharT, _Traits > &__x, basic_fstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_ifstream< _CharT, _Traits > &__x, basic_ifstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_ofstream< _CharT, _Traits > &__x, basic_ofstream< _CharT, _Traits > &__y)`

7.551.1 Detailed Description

This is a Standard C++ Library header.

7.552 functional File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_boyer_moore_searching`
- `#define __cpp_lib_experimental_not_fn`
- `#define _GLIBCXX_EXPERIMENTAL_FUNCTIONAL`

Typedefs

- `template<typename _RAIter, typename _Hash, typename _Pred, typename _Val = typename iterator_traits<_RAIter>::value_type, typename _Diff = typename iterator_traits<_RAIter>::difference_type>`
`using std::experimental::boyer_moore_base_t = std::conditional_t< std::is_byte_like< _Val, _Pred >::value, __boyer_moore_array_base< _Diff, 256, _Pred >, __boyer_moore_map_base< _Val, _Diff, _Hash, _Pred > >`

Functions

- `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>`
`boyer_moore_horspool_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::make_boyer_moore_horspool_searcher`
`(_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>`
`boyer_moore_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::make_boyer_moore_searcher`
`(_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _ForwardIterator, typename _BinaryPredicate = std::equal_to<>>`
`default_searcher< _ForwardIterator, _BinaryPredicate > std::experimental::make_default_searcher (_ForwardIterator __pat_first, _ForwardIterator __pat_last, _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _Fn >`
`auto std::experimental::not_fn (_Fn &&__fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn && >::value)`

Variables

- `template<typename _Tp >`
`constexpr bool std::experimental::is_bind_expression_v`
- `template<typename _Tp >`
`constexpr int std::experimental::is_placeholder_v`

7.552.1 Detailed Description

This is a TS C++ Library header.

7.553 functional File Reference

Classes

- class [__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >](#)
- struct [__gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::constant_unary_fun< _Result, _Argument >](#)
- struct [__gnu_cxx::constant_void_fun< _Result >](#)
- struct [__gnu_cxx::project1st< _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::project2nd< _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::select1st< _Pair >](#)
- struct [__gnu_cxx::select2nd< _Pair >](#)
- class [__gnu_cxx::subtractive_rng](#)
- class [__gnu_cxx::unary_compose< _Operation1, _Operation2 >](#)

Namespaces

- namespace [__gnu_cxx](#)

Macros

- `#define _EXT_FUNCTIONAL`

Functions

- [template<class _Operation1, class _Operation2 > __gnu_cxx::compose1](#) (const _Operation1 &__fn1, const _↵
Operation2 &__fn2)
- [template<class _Operation1, class _Operation2, class _Operation3 > __gnu_cxx::compose2](#) (const _Operation1 &__fn1,
const _Operation2 &__fn2, const _Operation3 &__fn3)
- [template<class _Result > __gnu_cxx::constant0](#) (const _Result &__val)
- [template<class _Result > __gnu_cxx::constant1](#) (const _Result &__val)
- [template<class _Result > __gnu_cxx::constant2](#) (const _Result &__val)
- [template<class _Tp > __gnu_cxx::identity_element](#) (std::multiplies< _Tp >)
- [template<class _Tp > __gnu_cxx::identity_element](#) (std::plus< _Tp >)
- [template<class _Ret, class _Tp, class _Arg > __gnu_cxx::mem_fun1](#) (_Ret(_Tp::*__f)(_Arg) const)
- [template<class _Ret, class _Tp, class _Arg > __gnu_cxx::mem_fun1](#) (_Ret(_Tp::*__f)(_Arg))
- [template<class _Ret, class _Tp, class _Arg > __gnu_cxx::mem_fun1_ref](#) (_Ret(_Tp::*__f)(_Arg) const)
- [template<class _Ret, class _Tp, class _Arg > __gnu_cxx::mem_fun1_ref](#) (_Ret(_Tp::*__f)(_Arg))

7.553.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.554 functional File Reference

Classes

- class `std::_Not_fn<_Fn>`
- struct `std::_Placeholder<_Num>`
- struct `std::is_bind_expression<_Tp>`
- struct `std::is_bind_expression<_Bind<_Signature>>`
- struct `std::is_bind_expression<_Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const _Bind<_Signature>>`
- struct `std::is_bind_expression<const _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<const volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_placeholder<_Tp>`
- struct `std::is_placeholder<_Placeholder<_Num>>`

Namespaces

- namespace `std`
- namespace `std::placeholders`

Macros

- `#define __cpp_lib_bind_front`
- `#define __cpp_lib_boyer_moore_searcher`
- `#define __cpp_lib_constexpr_functional`
- `#define __cpp_lib_invoke`
- `#define __cpp_lib_not_fn`
- `#define _GLIBCXX_FUNCTIONAL`
- `#define _GLIBCXX_NOT_FN_CALL_OP(_QUALS)`

Typedefs

- `template<typename _RAIter, typename _Hash, typename _Pred, typename _Val = typename iterator_traits<_RAIter>::value_type, typename _Diff = typename iterator_traits<_RAIter>::difference_type>`
`using std::_boyer_moore_base_t = __conditional_t<__is_byte_like<_Val, _Pred>::value, __boyer_moore_←`
`_array_base<_Diff, 256, _Pred>, __boyer_moore_map_base<_Val, _Diff, _Hash, _Pred>>`
- `template<typename _Tp, typename _Tp2 = typename decay<_Tp>::type>`
`using std::_is_socketlike = __or_<is_integral<_Tp2>, is_enum<_Tp2>>`
- `template<typename _Fn, typename... _Args>`
`using std::_Bind_front_t = _Bind_front<decay_t<_Fn>, decay_t<_Args>...>`

Functions

- `template<typename _Func, typename... _BoundArgs>`
`constexpr _Bind_helper<__is_socketlike<_Func>::value, _Func, _BoundArgs...>::type std::bind(_Func &&←`
`__f, _BoundArgs &&... __args)`
- `template<typename _Result, typename _Func, typename... _BoundArgs>`
`constexpr _Bindres_helper<_Result, _Func, _BoundArgs...>::type std::bind(_Func &&__f, _BoundArgs &&...←`
`__args)`

- `template<typename _Fn, typename... _Args>`
`constexpr _Bind_front_t< _Fn, _Args... > std::bind_front (_Fn &&__fn, _Args &&... __args) noexcept(is_↵`
`nothrow_constructible_v< _Bind_front_t< _Fn, _Args... >, int, _Fn, _Args... >)`
- `template<typename _Callable, typename... _Args>`
`constexpr invoke_result_t< _Callable, _Args... > std::invoke (_Callable &&__fn, _Args &&... __args)`
`noexcept(is_nothrow_invocable_v< _Callable, _Args... >)`
- `template<typename _Tp, typename _Class>`
`constexpr _Mem_fn< _Tp _Class::* > std::mem_fn (_Tp _Class::*__pm) noexcept`
- `template<typename _Fn >`
`constexpr auto std::not_fn (_Fn &&__fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn &&`
`>::value)`

Variables

- `const _Placeholder< 1 > std::placeholders::_1`
- `const _Placeholder< 10 > std::placeholders::_10`
- `const _Placeholder< 11 > std::placeholders::_11`
- `const _Placeholder< 12 > std::placeholders::_12`
- `const _Placeholder< 13 > std::placeholders::_13`
- `const _Placeholder< 14 > std::placeholders::_14`
- `const _Placeholder< 15 > std::placeholders::_15`
- `const _Placeholder< 16 > std::placeholders::_16`
- `const _Placeholder< 17 > std::placeholders::_17`
- `const _Placeholder< 18 > std::placeholders::_18`
- `const _Placeholder< 19 > std::placeholders::_19`
- `const _Placeholder< 2 > std::placeholders::_2`
- `const _Placeholder< 20 > std::placeholders::_20`
- `const _Placeholder< 21 > std::placeholders::_21`
- `const _Placeholder< 22 > std::placeholders::_22`
- `const _Placeholder< 23 > std::placeholders::_23`
- `const _Placeholder< 24 > std::placeholders::_24`
- `const _Placeholder< 25 > std::placeholders::_25`
- `const _Placeholder< 26 > std::placeholders::_26`
- `const _Placeholder< 27 > std::placeholders::_27`
- `const _Placeholder< 28 > std::placeholders::_28`
- `const _Placeholder< 29 > std::placeholders::_29`
- `const _Placeholder< 3 > std::placeholders::_3`
- `const _Placeholder< 4 > std::placeholders::_4`
- `const _Placeholder< 5 > std::placeholders::_5`
- `const _Placeholder< 6 > std::placeholders::_6`
- `const _Placeholder< 7 > std::placeholders::_7`
- `const _Placeholder< 8 > std::placeholders::_8`
- `const _Placeholder< 9 > std::placeholders::_9`
- `template<typename _Tp >`
`constexpr bool std::is_bind_expression_v`
- `template<typename _Tp >`
`constexpr int std::is_placeholder_v`

7.554.1 Detailed Description

This is a Standard C++ Library header.

7.555 future File Reference

Classes

- class `std::__basic_future< _Res >`
- class `std::future< _Res >`
- class `std::future< _Res & >`
- class `std::future< void >`
- class `std::future_error`
- struct `std::is_error_code_enum< future_errc >`
- class `std::packaged_task< _Res(_ArgTypes...)>`
- class `std::promise< _Res >`
- class `std::promise< _Res & >`
- class `std::promise< void >`
- class `std::shared_future< _Res >`
- class `std::shared_future< _Res & >`
- class `std::shared_future< void >`

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_FUTURE`

Enumerations

- enum class `std::future_errc` { `future_already_retrieved` , `promise_already_satisfied` , `no_state` , `broken_promise` }
- enum class `std::future_status` { `ready` , `timeout` , `deferred` }
- enum class `std::launch` { `async` , `deferred` }

Functions

- template<typename _Fn , typename... _Args>
`future< __async_result_of< _Fn, _Args... > > std::async (_Fn &&__fn, _Args &&... __args)`
- template<typename _Fn , typename... _Args>
`future< __async_result_of< _Fn, _Args... > > std::async (launch __policy, _Fn &&__fn, _Args &&... __args)`
- const `error_category & std::future_category ()` noexcept
- `error_code std::make_error_code (future_errc __errc)` noexcept
- `error_condition std::make_error_condition (future_errc __errc)` noexcept
- constexpr `launch std::operator& (launch __x, launch __y)` noexcept
- `launch & std::operator&= (launch &__x, launch __y)` noexcept
- constexpr `launch std::operator^ (launch __x, launch __y)` noexcept
- `launch & std::operator^= (launch &__x, launch __y)` noexcept
- constexpr `launch std::operator| (launch __x, launch __y)` noexcept
- `launch & std::operator|= (launch &__x, launch __y)` noexcept
- constexpr `launch std::operator~ (launch __x)` noexcept
- template<typename _Fun , typename _Signature = typename __function_guide_helper<decltype(&_Fun::operator())>::type>
`std::packaged_task (_Fun) -> packaged_task< _Signature >`
- template<typename _Res , typename... _ArgTypes>
`std::packaged_task (_Res(*)(_ArgTypes...)) -> packaged_task< _Res(_ArgTypes...)>`

- `template<typename _Res, typename... _ArgTypes>`
`void std::swap (packaged_task< _Res(_ArgTypes...)> &__x, packaged_task< _Res(_ArgTypes...)> &__y) noexcept`
- `template<typename _Res >`
`void std::swap (promise< _Res > &__x, promise< _Res > &__y) noexcept`

7.555.1 Detailed Description

This is a Standard C++ Library header.

7.556 iomanip File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_quoted_string_io`
- `#define _GLIBCXX_IOMANIP`

Functions

- `template<typename _MoneyT >`
`_Get_money< _MoneyT > std::get_money (_MoneyT &__mon, bool __intl=false)`
- `template<typename _CharT >`
`_Get_time< _CharT > std::get_time (std::tm *__tmb, const _CharT *__fmt)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_↵
money< _MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_time<
_CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _↵
Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setbase
__f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setfill<
_CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setiosflags
__f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _↵
Setprecision __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setw __f)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Get_money<
_MoneyT > __f)`

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Get_time<`
`_CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Resetiosflags`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Setbase __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Setfill< _CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Setiosflags`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Setprecision`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > & __is, _Setw __f)`
- `template<typename _MoneyT >`
`_Put_money< _MoneyT > std::put_money (const _MoneyT & __mon, bool __intl=false)`
- `template<typename _CharT >`
`_Put_time< _CharT > std::put_time (const std::tm * __tmb, const _CharT * __fmt)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto std::quoted (basic_string< _CharT, _Traits, _Alloc > & __string, _CharT __delim=_CharT(""), _CharT __escape =`
`_CharT("\\"))`
- `template<typename _CharT, typename _Traits >`
`auto std::quoted (basic_string_view< _CharT, _Traits > __sv, _CharT __delim=_CharT(""), _CharT __escape =`
`_CharT("\\"))`
- `template<typename _CharT >`
`auto std::quoted (const _CharT * __string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto std::quoted (const basic_string< _CharT, _Traits, _Alloc > & __string, _CharT __delim=_CharT(""), _CharT`
`__escape = _CharT("\\"))`
- `_Resetiosflags std::resetiosflags (ios_base::fmtflags __mask)`
- `_Setbase std::setbase (int __base)`
- `template<typename _CharT >`
`_Setfill< _CharT > std::setfill (_CharT __c)`
- `_Setiosflags std::setiosflags (ios_base::fmtflags __mask)`
- `_Setprecision std::setprecision (int __n)`
- `_Setw std::setw (int __n)`

7.556.1 Detailed Description

This is a Standard C++ Library header.

7.557 ios File Reference

Macros

- `#define _GLIBCXX_IOS`

7.557.1 Detailed Description

This is a Standard C++ Library header.

7.558 iosfwd File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_IOSFWD`

Typedefs

- typedef [basic_filebuf](#)< char > [std::filebuf](#)
- typedef [basic_fstream](#)< char > [std::fstream](#)
- typedef [basic_ifstream](#)< char > [std::ifstream](#)
- typedef [basic_ios](#)< char > [std::ios](#)
- typedef [basic_iostream](#)< char > [std::iostream](#)
- typedef [basic_istream](#)< char > [std::istream](#)
- typedef [basic_istreamstream](#)< char > [std::istreamstream](#)
- typedef [basic_ofstream](#)< char > [std::ofstream](#)
- typedef [basic_ostream](#)< char > [std::ostream](#)
- typedef [basic_ostreamstream](#)< char > [std::ostreamstream](#)
- typedef [basic_streambuf](#)< char > [std::streambuf](#)
- typedef [basic_stringbuf](#)< char > [std::stringbuf](#)
- typedef [basic_stringstream](#)< char > [std::stringstream](#)
- typedef [basic_filebuf](#)< wchar_t > [std::wfilebuf](#)
- typedef [basic_fstream](#)< wchar_t > [std::wfstream](#)
- typedef [basic_ifstream](#)< wchar_t > [std::wifstream](#)
- typedef [basic_ios](#)< wchar_t > [std::wios](#)
- typedef [basic_iostream](#)< wchar_t > [std::wiostream](#)
- typedef [basic_istream](#)< wchar_t > [std::wistream](#)
- typedef [basic_istreamstream](#)< wchar_t > [std::wistreamstream](#)
- typedef [basic_ofstream](#)< wchar_t > [std::wofstream](#)
- typedef [basic_ostream](#)< wchar_t > [std::wostream](#)
- typedef [basic_ostreamstream](#)< wchar_t > [std::wostreamstream](#)
- typedef [basic_streambuf](#)< wchar_t > [std::wstreambuf](#)
- typedef [basic_stringbuf](#)< wchar_t > [std::wstringbuf](#)
- typedef [basic_stringstream](#)< wchar_t > [std::wstringstream](#)

7.558.1 Detailed Description

This is a Standard C++ Library header.

7.559 iostream File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_IOSTREAM`

Variables

- static `ios_base::Init` [std::__ioinit](#)

Standard Stream Objects

The `<iostream>` header declares the eight standard stream objects. For other declarations, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html> and the [I/O forward declarations](#)

They are required by default to cooperate with the global C library's `FILE` streams, and to be available during program startup and termination. For more information, see the section of the manual linked to above.

- `ostream` [std::cerr](#)
- `istream` [std::cin](#)
- `ostream` [std::clog](#)
- `ostream` [std::cout](#)
- `wostream` [std::wcerr](#)
- `wistream` [std::wcin](#)
- `wostream` [std::wclog](#)
- `wostream` [std::wcout](#)

7.559.1 Detailed Description

This is a Standard C++ Library header.

7.560 istream File Reference

Classes

- class [std::basic_istream<_CharT, _Traits>](#)
- class [std::basic_istream<_CharT, _Traits>](#)
- class [std::basic_istream<_CharT, _Traits>::sentry](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_ISTREAM`

Typedefs

- `template<typename _Is, typename _Tp>`
using [std::__rvalue_stream_extraction_t](#) = `_Is &&`

Functions

- `template<typename _CharT, typename _Traits>`
void [std::__istream_extract](#) ([basic_istream<_CharT, _Traits>](#) &, `_CharT *`, [streamsize](#))
- void [std::__istream_extract](#) ([istream](#) &, `char *`, [streamsize](#))
- `template<typename _Istream, typename _Tp>`
`__rvalue_stream_extraction_t<_Istream, _Tp>` [std::operator>>](#) (`_Istream &&__is`, `_Tp &&__x`)

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::ws (basic_istream< _CharT, _Traits > &__is)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, signed char &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, unsigned char &__c)`
- `template<typename _CharT, typename _Traits, size_t _Num>`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT(&__s)[_Num])`
- `template<class _Traits, size_t _Num>`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, signed char(&__s)[_Num])`
- `template<class _Traits, size_t _Num>`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, unsigned char(&__s)[_Num])`

7.560.1 Detailed Description

This is a Standard C++ Library header.

7.561 iterator File Reference

Classes

- class `std::experimental::fundamentals_v2::ostream_joiner< _DelimT, _CharT, _Traits >`

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define __cpp_lib_experimental_ostream_joiner`
- `#define _GLIBCXX_EXPERIMENTAL_ITERATOR`

Functions

- `template<typename _CharT, typename _Traits, typename _DelimT >`
`ostream_joiner< decay_t< _DelimT >, _CharT, _Traits > std::experimental::make_ostream_joiner (basic_ostream< _CharT, _Traits > &__os, _DelimT &&__delimiter)`

7.561.1 Detailed Description

This is a TS C++ Library header.

7.562 iterator File Reference

Namespaces

- namespace `__gnu_cxx`

Macros

- `#define _EXT_ITERATOR`

Functions

- `template<typename _InputIterator, typename _Distance >`
`void __gnu_cxx::__distance (_InputIterator __first, _InputIterator __last, _Distance &__n, std::input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Distance >`
`void __gnu_cxx::__distance (_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance &__n, std::random_access_iterator_tag)`
- `template<typename _InputIterator, typename _Distance >`
`void __gnu_cxx::distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`

7.562.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.563 iterator File Reference

Macros

- `#define __cpp_lib_null_iterators`
- `#define _GLIBCXX_ITERATOR`

7.563.1 Detailed Description

This is a Standard C++ Library header.

7.564 latch File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_latch`
- `#define _GLIBCXX_LATCH`

7.564.1 Detailed Description

This is a Standard C++ Library header.

7.565 limits File Reference

Classes

- struct [std::__numeric_limits_base](#)
- struct [std::numeric_limits< _Tp >](#)
- struct [std::numeric_limits< bool >](#)
- struct [std::numeric_limits< char >](#)
- struct [std::numeric_limits< char16_t >](#)
- struct [std::numeric_limits< char32_t >](#)
- struct [std::numeric_limits< double >](#)
- struct [std::numeric_limits< float >](#)

- struct `std::numeric_limits< int >`
- struct `std::numeric_limits< long >`
- struct `std::numeric_limits< long double >`
- struct `std::numeric_limits< long long >`
- struct `std::numeric_limits< short >`
- struct `std::numeric_limits< signed char >`
- struct `std::numeric_limits< unsigned char >`
- struct `std::numeric_limits< unsigned int >`
- struct `std::numeric_limits< unsigned long >`
- struct `std::numeric_limits< unsigned long long >`
- struct `std::numeric_limits< unsigned short >`
- struct `std::numeric_limits< wchar_t >`

Namespaces

- namespace `std`

Macros

- `#define __glibcxx_digits(T)`
- `#define __glibcxx_digits10(T)`
- `#define __glibcxx_digits10_b(T, B)`
- `#define __glibcxx_digits_b(T, B)`
- `#define __glibcxx_double_has_denorm_loss`
- `#define __glibcxx_double_tinyness_before`
- `#define __glibcxx_double_traps`
- `#define __glibcxx_float_has_denorm_loss`
- `#define __glibcxx_float_tinyness_before`
- `#define __glibcxx_float_traps`
- `#define __glibcxx_integral_traps`
- `#define __glibcxx_long_double_has_denorm_loss`
- `#define __glibcxx_long_double_tinyness_before`
- `#define __glibcxx_long_double_traps`
- `#define __glibcxx_max(T)`
- `#define __glibcxx_max_b(T, B)`
- `#define __glibcxx_max_digits10(T)`
- `#define __glibcxx_min(T)`
- `#define __glibcxx_min_b(T, B)`
- `#define __glibcxx_signed(T)`
- `#define __glibcxx_signed_b(T, B)`
- `#define __INT_N(TYPE, BITSIZE, EXT, UEXT)`
- `#define __INT_N_201103(TYPE)`
- `#define __INT_N_U201103(TYPE)`
- `#define _GLIBCXX_NUMERIC_LIMITS`

Enumerations

- enum `std::float_denorm_style` { `std::denorm_indeterminate` , `std::denorm_absent` , `std::denorm_present` }
- enum `std::float_round_style` { `round_indeterminate` , `std::round_toward_zero` , `std::round_to_nearest` , `std::round_toward_infinity` , `std::round_toward_neg_infinity` }

7.565.1 Detailed Description

This is a Standard C++ Library header.

7.566 list File Reference

Classes

- class `std::__debug::list<_Tp, _Allocator>`

Namespaces

- namespace `__gnu_debug`
- namespace `std`
- namespace `std::__debug`

Macros

- `#define _GLIBCXX20_ONLY(__expr)`
- `#define _GLIBCXX_DEBUG_LIST`
- `#define _GLIBCXX_LIST_REMOVE_RETURN_TYPE_TAG`

Functions

- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
`std::__debug::list(_InputIterator, _InputIterator, _Allocator= _Allocator()) -> list<_ValT, _Allocator>`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>>`
`std::__debug::list(size_t, _Tp, _Allocator= _Allocator()) -> list<_Tp, _Allocator>`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator!= (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator< (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator<= (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator== (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator> (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator>= (const list<_Tp, _Alloc> &__lhs, const list<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`void std::__debug::swap(list<_Tp, _Alloc> &__lhs, list<_Tp, _Alloc> &__rhs) noexcept(!*conditional */)`

7.566.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.567 list File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- #define `_GLIBCXX_EXPERIMENTAL_LIST`

Typedefs

- template<typename `_Tp` >
using `std::experimental::fundamentals_v2::pmr::list` = `std::list`< `_Tp`, `polymorphic_allocator`< `_Tp` > >

Functions

- template<typename `_Tp`, typename `_Alloc`, typename `_Up` >
void `std::experimental::erase` (`list`< `_Tp`, `_Alloc` > &__cont, const `_Up` &__value)
- template<typename `_Tp`, typename `_Alloc`, typename `_Predicate` >
void `std::experimental::erase_if` (`list`< `_Tp`, `_Alloc` > &__cont, `_Predicate` __pred)

7.567.1 Detailed Description

This is a TS C++ Library header.

7.568 list File Reference**Namespaces**

- namespace `std`

Macros

- #define `__cpp_lib_erase_if`
- #define `_GLIBCXX_LIST`

Typedefs

- template<typename `_Tp` >
using `std::pmr::list` = `std::list`< `_Tp`, `polymorphic_allocator`< `_Tp` > >

Functions

- template<typename `_Tp`, typename `_Alloc`, typename `_Up` >
`list`< `_Tp`, `_Alloc` >::size_type `std::erase` (`list`< `_Tp`, `_Alloc` > &__cont, const `_Up` &__value)
- template<typename `_Tp`, typename `_Alloc`, typename `_Predicate` >
`list`< `_Tp`, `_Alloc` >::size_type `std::erase_if` (`list`< `_Tp`, `_Alloc` > &__cont, `_Predicate` __pred)

7.568.1 Detailed Description

This is a Standard C++ Library header.

7.569 locale File Reference**Macros**

- #define `_GLIBCXX_LOCALE`

7.569.1 Detailed Description

This is a Standard C++ Library header.

7.570 map File Reference

Namespaces

- namespace [std](#)
- namespace [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_MAP`

7.570.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.571 map File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_MAP`

Typedefs

- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>>>`
using `std::experimental::fundamentals_v2::pmr::map` = [std::map](#)< _Key, _Tp, _Compare, polymorphic_allocator< [pair](#)< const _Key, _Tp > > >
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>>>`
using `std::experimental::fundamentals_v2::pmr::multimap` = [std::multimap](#)< _Key, _Tp, _Compare, polymorphic_allocator< [pair](#)< const _Key, _Tp > > >

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
void `std::experimental::erase_if` ([map](#)< _Key, _Tp, _Compare, _Alloc > &__cont, _Predicate __pred)
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
void `std::experimental::erase_if` ([multimap](#)< _Key, _Tp, _Compare, _Alloc > &__cont, _Predicate __pred)

7.571.1 Detailed Description

This is a TS C++ Library header.

7.572 map File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_MAP`

Typedefs

- `template<typename _Key , typename _Tp , typename _Cmp = std::less<_Key>>
using std::pmr::map = std::map< _Key, _Tp, _Cmp, polymorphic_allocator< pair< const _Key, _Tp > > >`
- `template<typename _Key , typename _Tp , typename _Cmp = std::less<_Key>>
using std::pmr::multimap = std::multimap< _Key, _Tp, _Cmp, polymorphic_allocator< pair< const _Key, _Tp
> > >`

Functions

- `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc , typename _Predicate >
map< _Key, _Tp, _Compare, _Alloc >::size_type std::erase_if (map< _Key, _Tp, _Compare, _Alloc > &__cont,
_Predicate __pred)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc , typename _Predicate >
multimap< _Key, _Tp, _Compare, _Alloc >::size_type std::erase_if (multimap< _Key, _Tp, _Compare, _Alloc >
&__cont, _Predicate __pred)`

7.572.1 Detailed Description

This is a Standard C++ Library header.

7.573 math.h File Reference

7.573.1 Detailed Description

This is a Standard C++ Library header.

7.574 memory File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_observer_ptr`
- `#define _GLIBCXX_EXPERIMENTAL_MEMORY`

Functions

- `template<typename _Tp >
observer_ptr< _Tp > std::experimental::make_observer (_Tp *__p) noexcept`
- `template<typename _Tp >
bool std::experimental::operator!= (nullptr_t, observer_ptr< _Tp > __p) noexcept`
- `template<typename _Tp >
bool std::experimental::operator!= (observer_ptr< _Tp > __p, nullptr_t) noexcept`
- `template<typename _Tp , typename _Up >
bool std::experimental::operator!= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp , typename _Up >
bool std::experimental::operator< (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp , typename _Up >
bool std::experimental::operator<= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp >
bool std::experimental::operator== (nullptr_t, observer_ptr< _Tp > __p) noexcept`

- `template<typename _Tp >`
`bool std::experimental::operator== (observer_ptr< _Tp > __p, nullptr_t) noexcept`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::operator== (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::operator> (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::operator>= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp >`
`void std::experimental::swap (observer_ptr< _Tp > &__p1, observer_ptr< _Tp > &__p2) noexcept`

7.574.1 Detailed Description

This is a TS C++ Library header.

7.575 memory File Reference

Classes

- `struct __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >`

Namespaces

- namespace `__gnu_cxx`

Macros

- `#define _EXT_MEMORY`

Functions

- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __result)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __result, std::input_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Size, typename _ForwardIter >`
`std::pair< _RandomAccessIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_RandomAccessIter __first, _Size __count, _ForwardIter __result, std::random_access_iterator_tag)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Allocator >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter __result, _Allocator __alloc)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Tp >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter __result, std::allocator< _Tp >)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`std::pair< _InputIter, _ForwardIter > __gnu_cxx::uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __result)`

7.575.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGL STL subset).

7.576 memory File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_parallel_algorithm`
- `#define _GLIBCXX_MEMORY`

Enumerations

- enum class [std::pointer_safety](#) { relaxed , preferred , strict }

Functions

- void [std::declare_no_pointers](#) (char *, size_t)
- void [std::declare_reachable](#) (void *)
- [pointer_safety](#) [std::get_pointer_safety](#) () noexcept
- void [std::undeclare_no_pointers](#) (char *, size_t)
- template<typename _Tp >
_Tp * [std::undeclare_reachable](#) (_Tp * __p)

7.576.1 Detailed Description

This is a Standard C++ Library header.

7.577 memory_resource File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_memory_resources`
- `#define _GLIBCXX_EXPERIMENTAL_MEMORY_RESOURCE`

Typedefs

- template<typename _Alloc >
using [std::experimental::fundamentals_v2::pmr::resource_adaptor](#) = __resource_adaptor_imp< typename [allocator_traits](#)< _Alloc >::template rebind_alloc< char > >

Functions

- memory_resource * [std::experimental::fundamentals_v2::pmr::get_default_resource](#) () noexcept
- memory_resource * [std::experimental::fundamentals_v2::pmr::new_delete_resource](#) () noexcept
- memory_resource * [std::experimental::fundamentals_v2::pmr::null_memory_resource](#) () noexcept
- bool [std::experimental::fundamentals_v2::pmr::operator!=](#) (const memory_resource & __a, const memory_resource & __b) noexcept
- template<class _Tp1 , class _Tp2 >
bool [std::experimental::fundamentals_v2::pmr::operator!=](#) (const polymorphic_allocator< _Tp1 > & __a, const polymorphic_allocator< _Tp2 > & __b) noexcept

- `bool std::experimental::fundamentals_v2::pmr::operator==(const memory_resource &__a, const memory_resource &__b) noexcept`
- `template<class _Tp1, class _Tp2 >`
`bool std::experimental::fundamentals_v2::pmr::operator==(const polymorphic_allocator< _Tp1 > &__a, const polymorphic_allocator< _Tp2 > &__b) noexcept`
- `memory_resource * std::experimental::fundamentals_v2::pmr::set_default_resource (memory_resource *__r) noexcept`

7.577.1 Detailed Description

This is a TS C++ Library header.

7.577.2 Function Documentation

`get_default_resource()`

```
memory_resource * std::experimental::fundamentals_v2::pmr::get_default_resource ( ) [inline],
[noexcept]
```

Get the current default resource.

`set_default_resource()`

```
memory_resource * std::experimental::fundamentals_v2::pmr::set_default_resource (
    memory_resource * __r ) [inline], [noexcept]
```

Change the default resource and return the previous one.

7.578 memory_resource File Reference

Classes

- struct `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`
- class `std::pmr::memory_resource`
- struct `std::pmr::pool_options`
- class `std::pmr::synchronized_pool_resource`
- class `std::pmr::unsynchronized_pool_resource`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_memory_resource`
- `#define __cpp_lib_polymorphic_allocator`
- `#define _GLIBCXX_MEMORY_RESOURCE`

Functions

- `memory_resource * std::pmr::get_default_resource () noexcept`
- `memory_resource * std::pmr::new_delete_resource () noexcept`
- `memory_resource * std::pmr::null_memory_resource () noexcept`
- `bool std::pmr::operator==(const memory_resource &__a, const memory_resource &__b) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::pmr::operator==(const polymorphic_allocator< _Tp1 > &__a, const polymorphic_allocator< _Tp2 > &__b) noexcept`
- `memory_resource * std::pmr::set_default_resource (memory_resource *__r) noexcept`

7.578.1 Detailed Description

This is a Standard C++ Library header.

7.578.2 Function Documentation

get_default_resource()

`memory_resource * std::pmr::get_default_resource () [noexcept]`

Get the current default memory resource pointer.

new_delete_resource()

`memory_resource * std::pmr::new_delete_resource () [noexcept]`

A `pmr::memory_resource` that uses `new` to allocate memory.

null_memory_resource()

`memory_resource * std::pmr::null_memory_resource () [noexcept]`

A `pmr::memory_resource` that always throws `bad_alloc`

set_default_resource()

`memory_resource * std::pmr::set_default_resource (
memory_resource * __r) [noexcept]`

Replace the default memory resource pointer.

7.579 mutex File Reference

Classes

- struct `std::once_flag`
- class `std::recursive_mutex`
- class `std::recursive_timed_mutex`
- class `std::scoped_lock<_MutexTypes >`
- class `std::timed_mutex`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_scoped_lock`
- `#define _GLIBCXX_MUTEX`

Functions

- void `std::__once_proxy` (void)
- `template<typename _Callable, typename... _Args>`
void `std::call_once` (`once_flag` &__once, `_Callable` &&__f, `_Args` &&... __args)
- `template<typename _L1, typename _L2, typename... _L3>`
void `std::lock` (`_L1` &__l1, `_L2` &__l2, `_L3` &... __l3)
- `template<typename _L1, typename _L2, typename... _L3>`
int `std::try_lock` (`_L1` &__l1, `_L2` &__l2, `_L3` &... __l3)

7.579.1 Detailed Description

This is a Standard C++ Library header.

7.580 numbers File Reference

Namespaces

- namespace [std](#)
- namespace [std::numbers](#)

Macros

- `#define __cpp_lib_math_constants`
- `#define _GLIBCXX_NUMBERS`

Variables

- constexpr double **std::numbers::e**
- template<typename _Tp >
constexpr _Tp [std::numbers::e_v](#)
- constexpr double **std::numbers::egamma**
- template<typename _Tp >
constexpr _Tp [std::numbers::egamma_v](#)
- constexpr double **std::numbers::inv_pi**
- template<typename _Tp >
constexpr _Tp [std::numbers::inv_pi_v](#)
- constexpr double **std::numbers::inv_sqrt3**
- template<typename _Tp >
constexpr _Tp [std::numbers::inv_sqrt3_v](#)
- constexpr double **std::numbers::inv_sqrtpi**
- template<typename _Tp >
constexpr _Tp [std::numbers::inv_sqrtpi_v](#)
- constexpr double **std::numbers::ln10**
- template<typename _Tp >
constexpr _Tp [std::numbers::ln10_v](#)
- constexpr double **std::numbers::ln2**
- template<typename _Tp >
constexpr _Tp [std::numbers::ln2_v](#)
- constexpr double **std::numbers::log10e**
- template<typename _Tp >
constexpr _Tp [std::numbers::log10e_v](#)
- constexpr double **std::numbers::log2e**
- template<typename _Tp >
constexpr _Tp [std::numbers::log2e_v](#)
- constexpr double **std::numbers::phi**
- template<typename _Tp >
constexpr _Tp [std::numbers::phi_v](#)
- constexpr double **std::numbers::pi**
- template<typename _Tp >
constexpr _Tp [std::numbers::pi_v](#)
- constexpr double **std::numbers::sqrt2**
- template<typename _Tp >
constexpr _Tp [std::numbers::sqrt2_v](#)

- constexpr double **std::numbers::sqrt3**
- template<typename _Tp >
constexpr _Tp **std::numbers::sqrt3_v**

7.580.1 Detailed Description

This is a Standard C++ Library header.

7.581 numeric File Reference

Namespaces

- namespace **std**
- namespace **std::experimental**

Macros

- #define **__cpp_lib_experimental_gcd_lcm**
- #define **__GLIBCXX_EXPERIMENTAL_NUMERIC**

Functions

- template<typename _Mn, typename _Nn >
constexpr **common_type_t**< _Mn, _Nn > **std::experimental::gcd** (_Mn __m, _Nn __n) noexcept
- template<typename _Mn, typename _Nn >
constexpr **common_type_t**< _Mn, _Nn > **std::experimental::lcm** (_Mn __m, _Nn __n)

7.581.1 Detailed Description

This is a TS C++ Library header.

7.582 numeric File Reference

Namespaces

- namespace **__gnu_cxx**

Macros

- #define **_EXT_NUMERIC**

Functions

- template<typename _Tp, typename _Integer >
_Tp **__gnu_cxx::__power** (_Tp __x, _Integer __n)
- template<typename _Tp, typename _Integer, typename _MonoidOperation >
_Tp **__gnu_cxx::__power** (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)
- template<typename _Tp, typename _Integer >
_Tp **__gnu_cxx::power** (_Tp __x, _Integer __n)
- template<typename _Tp, typename _Integer, typename _MonoidOperation >
_Tp **__gnu_cxx::power** (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)

7.582.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

7.583 numeric File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_constexpr_numeric`
- `#define __cpp_lib_gcd`
- `#define __cpp_lib_gcd_lcm`
- `#define __cpp_lib_interpolate`
- `#define __cpp_lib_lcm`
- `#define __cpp_lib_parallel_algorithm`
- `#define _GLIBCXX_NUMERIC`
- `#define _PSTL_NUMERIC_FORWARD_DECLARED`

Functions

- `template<typename _Res, typename _Tp >`
`constexpr _Res std::__detail::__abs_r (_Tp __val)`
- `template<typename >`
`void std::__detail::__abs_r (bool)=delete`
- `template<typename _Tp >`
`constexpr _Tp std::__detail::__gcd (_Tp __m, _Tp __n)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`constexpr _OutputIterator std::exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __↵
result, _Tp __init)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _OutputIterator std::exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __↵
result, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > std::gcd (_Mn __m, _Nn __n) noexcept`
- `template<typename _InputIterator, typename _OutputIterator >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __↵
result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __↵
result, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _Tp >`
`constexpr _OutputIterator std::inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __↵
result, _BinaryOperation __binary_op, _Tp __init)`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > std::lcm (_Mn __m, _Nn __n) noexcept`
- `template<typename _Tp >`
`constexpr enable_if_t< is_object_v< _Tp >, _Tp * > std::midpoint (_Tp *__a, _Tp *__b) noexcept`
- `template<typename _Tp >`
`constexpr enable_if_t< __and_v< is_arithmetic< _Tp >, is_same< remove_cv_t< _Tp >, _Tp >, __not<
is_same< _Tp, bool > >, _Tp > std::midpoint (_Tp __a, _Tp __b) noexcept`
- `template<typename _InputIterator >`
`constexpr iterator_traits< _InputIterator >::value_type std::reduce (_InputIterator __first, _InputIterator __last)`
- `template<typename _InputIterator, typename _Tp >`
`constexpr _Tp std::reduce (_InputIterator __first, _InputIterator __last, _Tp __init)`

- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`constexpr _Tp std::reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform_exclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Tp __init, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _OutputIterator std::transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation, typename _UnaryOperation, typename _Tp >`
`constexpr _OutputIterator std::transform_inclusive_scan (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op, _UnaryOperation __unary_op, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation, typename _UnaryOperation >`
`constexpr _Tp std::transform_reduce (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`constexpr _Tp std::transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2 >`
`constexpr _Tp std::transform_reduce (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2)`

7.583.1 Detailed Description

This is a Standard C++ Library header.

7.584 numeric File Reference

Namespaces

- namespace [std](#)
- namespace [std::__parallel](#)

Macros

- `#define _GLIBCXX_PARALLEL_NUMERIC_H`

Functions

- `template<typename __RAIter, typename _Tp, typename _BinaryOperation >`
`_Tp std::__parallel::__accumulate_switch (__RAIter __begin, __RAIter __end, _Tp __init, _BinaryOperation __binary_op, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation, typename _IteratorTag >`
`_Tp std::__parallel::__accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, _IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Tp std::__parallel::__accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _IteratorTag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator std::__parallel::__adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::__adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, random_access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`

- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _IterTag1, typename _IterTag2 >`
`_Tp std::parallel::inner_product_switch (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::parallel::inner_product_switch (_RAIter1, _RAIter1, _RAIter2, _Tp, _BinaryFunction1, _BinaryFunction2, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IterTag1, typename _IterTag2 >`
`_OutputIterator std::parallel::partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::parallel::accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::parallel::accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::parallel::accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::parallel::accumulate (_Iter, _Iter, _Tp)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::parallel::accumulate (_Iter, _Iter, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::parallel::accumulate (_Iter, _Iter, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2)`

- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 > _Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::_Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 > _Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator > _OutputIterator std::__parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator > _OutputIterator std::__parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation > _OutputIterator std::__parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation > _OutputIterator std::__parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op)`

7.584.1 Detailed Description

Parallel STL function calls corresponding to `stl_numeric.h`. The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

7.585 optional File Reference

Classes

- class `std::experimental::fundamentals_v1::bad_optional_access`
- struct `std::hash< experimental::optional< _Tp > >`
- struct `std::experimental::fundamentals_v1::in_place_t`
- struct `std::experimental::fundamentals_v1::nullopt_t`
- class `std::experimental::fundamentals_v1::optional< _Tp >`

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define __cpp_lib_experimental_optional`
- `#define _GLIBCXX_EXPERIMENTAL_OPTIONAL`

Variables

- constexpr `in_place_t std::experimental::in_place`
- constexpr `nullopt_t std::experimental::nullopt`

7.585.1 Detailed Description

This is a TS C++ Library header.

7.586 optional File Reference

Classes

- struct `std::_Optional_base< _Tp, bool, bool >`
- class `std::bad_optional_access`
- struct `std::nullopt_t`
- class `std::optional< _Tp >`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_optional`
- `#define _GLIBCXX_OPTIONAL`

Typedefs

- `template<typename _Tp, typename _Up >`
`using std::__assigns_from_optional = __or_< is_assignable< _Tp &, const optional< _Up > & >,`
`is_assignable< _Tp &, optional< _Up > & >, is_assignable< _Tp &, const optional< _Up > && >,`
`is_assignable< _Tp &, optional< _Up > && >`
- `template<typename _Tp, typename _Up >`
`using std::__converts_from_optional = __or_< is_constructible< _Tp, const optional< _Up > & >,`
`is_constructible< _Tp, optional< _Up > & >, is_constructible< _Tp, const optional< _Up > && >,`
`is_constructible< _Tp, optional< _Up > && >, is_convertible< const optional< _Up > &, _Tp >, is_convertible<`
`optional< _Up > &, _Tp >, is_convertible< const optional< _Up > &&, _Tp >, is_convertible< optional< _Up`
`> &&, _Tp >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_eq_t = __optional_relop_t< decltype(std::declval< const _Tp >())==std::declval< const`
`_Up & >()) >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_ge_t = __optional_relop_t< decltype(std::declval< const _Tp & >())>=std::declval<`
`const _Up & >()) >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_gt_t = __optional_relop_t< decltype(std::declval< const _Tp & >()) > std::declval< const`
`_Up & >()) >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_le_t = __optional_relop_t< decltype(std::declval< const _Tp & >())<=std::declval< const`
`_Up & >()) >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_lt_t = __optional_relop_t< decltype(std::declval< const _Tp & >())< std::declval< const`
`_Up & >()) >`
- `template<typename _Tp, typename _Up >`
`using std::__optional_ne_t = __optional_relop_t< decltype(std::declval< const _Tp & >())!=std::declval< const`
`_Up & >()) >`
- `template<typename _Tp >`
`using std::__optional_relop_t = enable_if_t< is_convertible< _Tp, bool >::value, bool >`

Functions

- void **std::__throw_bad_optional_access** ()
- template<typename _Tp, typename... _Args>
constexpr **enable_if_t**< is_constructible_v< _Tp, _Args... >, **optional**< _Tp > > **std::make_optional** (_Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, _Args... >)
- template<typename _Tp >
constexpr **enable_if_t**< is_constructible_v< **decay_t**< _Tp >, _Tp >, **optional**< **decay_t**< _Tp > > > **std::make_optional** (_Tp &&__t) noexcept(is_nothrow_constructible_v< **optional**< **decay_t**< _Tp > >, _Tp >)
- template<typename _Tp, typename _Up, typename... _Args>
constexpr **enable_if_t**< is_constructible_v< _Tp, **initializer_list**< _Up > &, _Args... >, **optional**< _Tp > > **std::make_optional** (**initializer_list**< _Up > __il, _Args &&... __args) noexcept(is_nothrow_constructible_v< _Tp, **initializer_list**< _Up > &, _Args... >)
- template<typename _Tp, typename _Up >
constexpr auto **std::operator!=** (const _Up &__lhs, const **optional**< _Tp > &__rhs) -> __optional_ne_t< _Up, _Tp >
- template<typename _Tp, typename _Up >
constexpr auto **std::operator!=** (const **optional**< _Tp > &__lhs, const _Up &__rhs) -> __optional_ne_t< _Tp, _Up >
- template<typename _Tp, typename _Up >
constexpr auto **std::operator!=** (const **optional**< _Tp > &__lhs, const **optional**< _Up > &__rhs) -> __optional_ne_t< _Tp, _Up >
- template<typename _Tp >
constexpr bool **std::operator!=** (const **optional**< _Tp > &__lhs, **nullopt_t**) noexcept
- template<typename _Tp >
constexpr bool **std::operator!=** (**nullopt_t**, const **optional**< _Tp > &__rhs) noexcept
- template<typename _Tp, typename _Up >
constexpr auto **std::operator**< (const _Up &__lhs, const **optional**< _Tp > &__rhs) -> __optional_lt_t< _Up, _Tp >
- template<typename _Tp >
constexpr bool **std::operator**< (const **optional**< _Tp > &, **nullopt_t**) noexcept
- template<typename _Tp, typename _Up >
constexpr auto **std::operator**< (const **optional**< _Tp > &__lhs, const _Up &__rhs) -> __optional_lt_t< _Tp, _Up >
- template<typename _Tp, typename _Up >
constexpr auto **std::operator**< (const **optional**< _Tp > &__lhs, const **optional**< _Up > &__rhs) -> __optional_lt_t< _Tp, _Up >
- template<typename _Tp >
constexpr bool **std::operator**< (**nullopt_t**, const **optional**< _Tp > &__rhs) noexcept
- template<typename _Tp, typename _Up >
constexpr auto **std::operator<=** (const _Up &__lhs, const **optional**< _Tp > &__rhs) -> __optional_le_t< _Up, _Tp >
- template<typename _Tp, typename _Up >
constexpr auto **std::operator<=** (const **optional**< _Tp > &__lhs, const _Up &__rhs) -> __optional_le_t< _Tp, _Up >
- template<typename _Tp, typename _Up >
constexpr auto **std::operator<=** (const **optional**< _Tp > &__lhs, const **optional**< _Up > &__rhs) -> __optional_le_t< _Tp, _Up >
- template<typename _Tp >
constexpr bool **std::operator<=** (const **optional**< _Tp > &__lhs, **nullopt_t**) noexcept
- template<typename _Tp >
constexpr bool **std::operator<=** (**nullopt_t**, const **optional**< _Tp > &) noexcept

- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_eq_t< _Up, _Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator== (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_eq_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator== (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::operator== (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_gt_t< _Up, _Tp >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator> (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_gt_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator> (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::operator> (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const _Up &__lhs, const optional< _Tp > &__rhs) -> __optional_ge_t< _Up, _Tp >`
- `template<typename _Tp >`
`constexpr bool std::operator>= (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const optional< _Tp > &__lhs, const _Up &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename _Tp, typename _Up >`
`constexpr auto std::operator>= (const optional< _Tp > &__lhs, const optional< _Up > &__rhs) -> __optional_ge_t< _Tp, _Up >`
- `template<typename _Tp >`
`constexpr bool std::operator>= (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp >`
`std::optional (_Tp) -> optional< _Tp >`
- `template<typename _Tp >`
`enable_if_t<!(is_move_constructible_v< _Tp > &&is_swappable_v< _Tp >)> std::swap (optional< _Tp > &, optional< _Tp > &)=delete`
- `template<typename _Tp >`
`constexpr enable_if_t< is_move_constructible_v< _Tp > &&is_swappable_v< _Tp > > std::swap (optional< _Tp > &__lhs, optional< _Tp > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))`

Variables

- `template<typename _Tp >`
`constexpr bool std::__is_optional_v`

- `template<typename _Tp >`
`constexpr bool std::__is_optional_v< optional< _Tp > >`
- `constexpr nullopt_t std::nullopt`

7.586.1 Detailed Description

This is a Standard C++ Library header.

7.587 ostream File Reference

Classes

- class [std::basic_ostream< _CharT, _Traits >](#)
- class [std::basic_ostream< _CharT, _Traits >::sentry](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_OSTREAM`

Typedefs

- `template<typename _Os, typename _Tp >`
`using std::__rvalue_stream_insertion_t = _Os &&`

Functions

- `template<typename _CharT, typename _Traits >`
[basic_ostream](#)< _CharT, _Traits > & **std::endl** ([basic_ostream](#)< _CharT, _Traits > &__os)
- `template<typename _CharT, typename _Traits >`
[basic_ostream](#)< _CharT, _Traits > & **std::ends** ([basic_ostream](#)< _CharT, _Traits > &__os)
- `template<typename _CharT, typename _Traits >`
[basic_ostream](#)< _CharT, _Traits > & **std::flush** ([basic_ostream](#)< _CharT, _Traits > &__os)
- `template<typename _Ostream, typename _Tp >`
`__rvalue_stream_insertion_t< _Ostream, _Tp > std::operator<< (_Ostream &&__os, const _Tp &__x)`
- `template<typename _CharT, typename _Traits >`
[basic_ostream](#)< _CharT, _Traits > & **std::operator<<** ([basic_ostream](#)< _CharT, _Traits > &__out, _CharT __c)
- `template<typename _CharT, typename _Traits >`
[basic_ostream](#)< _CharT, _Traits > & **std::operator<<** ([basic_ostream](#)< _CharT, _Traits > &__out, char __c)
- `template<typename _Traits >`
[basic_ostream](#)< char, _Traits > & **std::operator<<** ([basic_ostream](#)< char, _Traits > &, char16_t)=delete
- `template<typename _Traits >`
[basic_ostream](#)< char, _Traits > & **std::operator<<** ([basic_ostream](#)< char, _Traits > &, char32_t)=delete
- `template<typename _Traits >`
[basic_ostream](#)< char, _Traits > & **std::operator<<** ([basic_ostream](#)< char, _Traits > &, wchar_t)=delete
- `template<typename _Traits >`
[basic_ostream](#)< char, _Traits > & **std::operator<<** ([basic_ostream](#)< char, _Traits > &__out, char __c)
- `template<typename _Traits >`
[basic_ostream](#)< char, _Traits > & **std::operator<<** ([basic_ostream](#)< char, _Traits > &__out, signed char __c)

- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, unsigned char __c)`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & std::operator<< (basic_ostream< wchar_t, _Traits > &, char16_t)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & std::operator<< (basic_ostream< wchar_t, _Traits > &, char32_t)=delete`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, const _CharT *__s)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, const char *__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &, const char16_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &, const char32_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &, const wchar_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const char *__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const signed char *__s)`
- `template<typename _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const unsigned char *__s)`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & std::operator<< (basic_ostream< wchar_t, _Traits > &, const char16_t *)=delete`
- `template<typename _Traits >`
`basic_ostream< wchar_t, _Traits > & std::operator<< (basic_ostream< wchar_t, _Traits > &, const char32_t *)=delete`

7.587.1 Detailed Description

This is a Standard C++ Library header.

7.588 algo.h File Reference

Classes

- struct `std::__parallel::CRandNumber< _MustBeInt >`

Namespaces

- namespace `std`
- namespace `std::__parallel`

Functions

- `template<typename _Filterator, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator std::parallel::adjacent_find_switch (_Filterator __begin, _Filterator __end, _BinaryPredicate __pred, _IteratorTag)`
- `template<typename _Filterator, typename _IteratorTag >`
`_Filterator std::parallel::adjacent_find_switch (_Filterator __begin, _Filterator __end, _IteratorTag)`
- `template<typename _RAlter, typename _BinaryPredicate >`
`_RAlter std::parallel::adjacent_find_switch (_RAlter __begin, _RAlter __end, _BinaryPredicate __pred, random_access_iterator_tag)`
- `template<typename _RAlter >`
`_RAlter std::parallel::adjacent_find_switch (_RAlter __begin, _RAlter __end, random_access_iterator_tag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type std::parallel::count_if_switch (_Iter __begin, _Iter __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAlter, typename _Predicate >`
`iterator_traits< _RAlter >::difference_type std::parallel::count_if_switch (_RAlter __begin, _RAlter __end, _Predicate __pred, random_access_iterator_tag, _gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type std::parallel::count_switch (_Iter __begin, _Iter __end, const _Tp & __value, _IteratorTag)`
- `template<typename _RAlter, typename _Tp >`
`iterator_traits< _RAlter >::difference_type std::parallel::count_switch (_RAlter __begin, _RAlter __end, const _Tp & __value, random_access_iterator_tag, _gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Filterator, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter std::parallel::find_first_of_switch (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2, _BinaryPredicate __comp, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _Filterator, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter std::parallel::find_first_of_switch (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAlter, typename _Filterator, typename _BinaryPredicate, typename _IteratorTag >`
`_RAlter std::parallel::find_first_of_switch (_RAlter __begin1, _RAlter __end1, _Filterator __begin2, _Filterator __end2, _BinaryPredicate __comp, random_access_iterator_tag, _IteratorTag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`_Iter std::parallel::find_if_switch (_Iter __begin, _Iter __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAlter, typename _Predicate >`
`_RAlter std::parallel::find_if_switch (_RAlter __begin, _RAlter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Iter std::parallel::find_switch (_Iter __begin, _Iter __end, const _Tp & __val, _IteratorTag)`
- `template<typename _RAlter, typename _Tp >`
`_RAlter std::parallel::find_switch (_RAlter __begin, _RAlter __end, const _Tp & __val, random_access_iterator_tag)`
- `template<typename _Iter, typename _Function, typename _IteratorTag >`
`_Function std::parallel::for_each_switch (_Iter __begin, _Iter __end, _Function __f, _IteratorTag)`
- `template<typename _RAlter, typename _Function >`
`_Function std::parallel::for_each_switch (_RAlter __begin, _RAlter __end, _Function __f, random_access_iterator_tag, _gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator, typename _IteratorTag >`
`_OutputIterator std::parallel::generate_n_switch (_OutputIterator __begin, _Size __n, _Generator __gen, _IteratorTag)`
- `template<typename _RAlter, typename _Size, typename _Generator >`
`_RAlter std::parallel::generate_n_switch (_RAlter __begin, _Size __n, _Generator __gen, random_access_iterator_tag, _gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Generator, typename _IteratorTag >`
`void std::parallel::generate_switch (_Filterator __begin, _Filterator __end, _Generator __gen, _IteratorTag)`

- `template<typename _RAIter, typename _Generator >`
`void std::__parallel::__generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator std::__parallel::__max_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IteratorTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::__parallel::__max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::__parallel::__merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__parallel::__merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator std::__parallel::__min_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IteratorTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::__parallel::__min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Predicate, typename _IteratorTag >`
`_Filterator std::__parallel::__partition_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::__parallel::__partition_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp, typename _IteratorTag >`
`void std::__parallel::__replace_if_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp & __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate, typename _Tp >`
`void std::__parallel::__replace_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, const _Tp & __new_value, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Tp, typename _IteratorTag >`
`void std::__parallel::__replace_switch (_Filterator __begin, _Filterator __end, const _Tp & __old_value, const _Tp & __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`void std::__parallel::__replace_switch (_RAIter __begin, _RAIter __end, const _Tp & __old_value, const _Tp & __new_value, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator std::__parallel::__search_n_switch (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred, _IteratorTag)`
- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_RAIter std::__parallel::__search_n_switch (_RAIter __begin, _RAIter __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred, random_access_iterator_tag)`
- `template<typename _Filterator1, typename _Filterator2, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Filterator1 std::__parallel::__search_switch (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _BinaryPredicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _Filterator1, typename _Filterator2, typename _IteratorTag1, typename _IteratorTag2 >`
`_Filterator1 std::__parallel::__search_switch (_Filterator1 __begin1, _Filterator1 __end1, _Filterator2 __begin2, _Filterator2 __end2, _IteratorTag1, _IteratorTag2)`

- `template<typename _RAIter1, typename _RAIter2, typename _BinaryPredicate >`
`_RAIter1 std:: parallel:: search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2`
`__end2, _BinaryPredicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2 >`
`_RAIter1 std:: parallel:: search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2`
`__end2, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std:: parallel:: set_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2,`
`_IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std:: parallel:: set_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2`
`__begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std:: parallel:: set_intersection_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2,`
`_IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std:: parallel:: set_intersection_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2`
`__begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std:: parallel:: set_symmetric_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2`
`__begin2, _IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std:: parallel:: set_symmetric_difference_switch (_RAIter1 __begin1, _RAIter1 __end1,`
`_RAIter2 __begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std:: parallel:: set_union_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2`
`__end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std:: parallel:: set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2`
`__begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_RAIter2 std:: parallel:: transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation`
`__unary_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation >`
`_RAIter2 std:: parallel:: transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation`
`__unary_op, random_access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism`
`parallelism_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _BinaryOperation, typename _Tag1, typename _Tag2, typename _Tag3 >`
`_OutputIterator std:: parallel:: transform2_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _OutputIterator`
`__result, _BinaryOperation __binary_op, _Tag1, _Tag2, _Tag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BinaryOperation >`
`_RAIter3 std:: parallel:: transform2_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter3`
`__result, _BinaryOperation __binary_op, random_access_iterator_tag, random_access_iterator_tag,
random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`

- `template<typename _Iter, typename _OutputIterator, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator std::__parallel::__unique_copy_switch` (`_Iter __begin`, `_Iter __last`, `_OutputIterator __out`, `↵`
`_Predicate __pred`, `_IteratorTag1`, `_IteratorTag2`)
- `template<typename _RAIter, typename _RandomAccessOutputIterator, typename _Predicate >`
`_RandomAccessOutputIterator std::__parallel::__unique_copy_switch` (`_RAIter __begin`, `_RAIter __last`, `↵`
`RandomAccessOutputIterator __out`, `_Predicate __pred`, `random_access_iterator_tag`, `random_access_iterator_tag`)
- `template<typename _FIterator >`
`_FIterator std::__parallel::adjacent_find` (`_FIterator __begin`, `_FIterator __end`)
- `template<typename _FIterator >`
`_FIterator std::__parallel::adjacent_find` (`_FIterator __begin`, `_FIterator __end`, `gnu_parallel::sequential_tag`)
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator std::__parallel::adjacent_find` (`_FIterator __begin`, `_FIterator __end`, `_BinaryPredicate __binary`, `↵`
`pred`, `gnu_parallel::sequential_tag`)
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator std::__parallel::adjacent_find` (`_FIterator __begin`, `_FIterator __end`, `_BinaryPredicate __pred`)
- `template<typename _Iter, typename _Tp >`
`iterator_traits<_Iter>::difference_type std::__parallel::count` (`_Iter __begin`, `_Iter __end`, `const _Tp &`, `↵`
`value`)
- `template<typename _Iter, typename _Tp >`
`iterator_traits<_Iter>::difference_type std::__parallel::count` (`_Iter __begin`, `_Iter __end`, `const _Tp &`, `↵`
`value`, `gnu_parallel::Parallelism`, `parallelism_tag`)
- `template<typename _Iter, typename _Tp >`
`iterator_traits<_Iter>::difference_type std::__parallel::count` (`_Iter __begin`, `_Iter __end`, `const _Tp &`, `↵`
`value`, `gnu_parallel::sequential_tag`)
- `template<typename _Iter, typename _Predicate >`
`iterator_traits<_Iter>::difference_type std::__parallel::count_if` (`_Iter __begin`, `_Iter __end`, `_Predicate __`, `↵`
`pred`)
- `template<typename _Iter, typename _Predicate >`
`iterator_traits<_Iter>::difference_type std::__parallel::count_if` (`_Iter __begin`, `_Iter __end`, `_Predicate __`, `↵`
`pred`, `gnu_parallel::Parallelism`, `parallelism_tag`)
- `template<typename _Iter, typename _Predicate >`
`iterator_traits<_Iter>::difference_type std::__parallel::count_if` (`_Iter __begin`, `_Iter __end`, `_Predicate __`, `↵`
`pred`, `gnu_parallel::sequential_tag`)
- `template<typename _Iter, typename _Tp >`
`_Iter std::__parallel::find` (`_Iter __begin`, `_Iter __end`, `const _Tp &`, `__val`)
- `template<typename _Iter, typename _Tp >`
`_Iter std::__parallel::find` (`_Iter __begin`, `_Iter __end`, `const _Tp &`, `__val`, `gnu_parallel::sequential_tag`)
- `template<typename _Iter, typename _FIterator >`
`_Iter std::__parallel::find_first_of` (`_Iter __begin1`, `_Iter __end1`, `_FIterator __begin2`, `_FIterator __end2`)
- `template<typename _Iter, typename _FIterator >`
`_Iter std::__parallel::find_first_of` (`_Iter __begin1`, `_Iter __end1`, `_FIterator __begin2`, `_FIterator __end2`, `↵`
`gnu_parallel::sequential_tag`)
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter std::__parallel::find_first_of` (`_Iter __begin1`, `_Iter __end1`, `_FIterator __begin2`, `_FIterator __end2`, `↵`
`BinaryPredicate __comp`)
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter std::__parallel::find_first_of` (`_Iter __begin1`, `_Iter __end1`, `_FIterator __begin2`, `_FIterator __end2`, `↵`
`BinaryPredicate __comp`, `gnu_parallel::sequential_tag`)
- `template<typename _Iter, typename _Predicate >`
`_Iter std::__parallel::find_if` (`_Iter __begin`, `_Iter __end`, `_Predicate __pred`)
- `template<typename _Iter, typename _Predicate >`
`_Iter std::__parallel::find_if` (`_Iter __begin`, `_Iter __end`, `_Predicate __pred`, `gnu_parallel::sequential_tag`)

- `template<typename _Iter, typename _Function >`
`_Function std::__parallel::for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function std::__parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f)`
- `template<typename _Iterator, typename _Function >`
`_Function std::__parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::Parallelism
__parallelism_tag)`
- `template<typename _Filterator, typename _Generator >`
`void std::__parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen)`
- `template<typename _Filterator, typename _Generator >`
`void std::__parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen, __gnu_parallel::Parallelism
__parallelism_tag)`
- `template<typename _Filterator, typename _Generator >`
`void std::__parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen, __gnu_parallel::sequential_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::__parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::__parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen,
__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::__parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen,
__gnu_parallel::sequential_tag)`
- `template<typename _Filterator >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end)`
- `template<typename _Filterator >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp,
__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::__parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp,
__gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::__parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag,
__parallelism_tag, _OutputIterator __result)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::__parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag,
__parallelism_tag, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag,
__parallelism_tag, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag,
__parallelism_tag, _OutputIterator __result, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator >`
`_Filterator std::__parallel::min_element (_Filterator __begin, _Filterator __end)`

- `template<typename _FIterator >`
`_FIterator std::__parallel::min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::Parallelism __↵`
`__parallelism_tag)`
- `template<typename _FIterator >`
`_FIterator std::__parallel::min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator std::__parallel::min_element (_FIterator __begin, _FIterator __end, _Compare __comp)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator std::__parallel::min_element (_FIterator __begin, _FIterator __end, _Compare __comp,`
`__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator std::__parallel::min_element (_FIterator __begin, _FIterator __end, _Compare __comp,`
`__gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp,`
`__gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp,`
`__gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Predicate >`
`_FIterator std::__parallel::partition (_FIterator __begin, _FIterator __end, _Predicate __pred)`
- `template<typename _FIterator, typename _Predicate >`
`_FIterator std::__parallel::partition (_FIterator __begin, _FIterator __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &__rand,`
`__gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__↵`
`new_value)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__↵`
`new_value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__↵`
`new_value, __gnu_parallel::sequential_tag)`

- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new↵`
`__value)`
- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new↵`
`__value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new↵`
`__value, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 std::parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2`
`__end2)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 std::parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2`
`__end2, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 std::parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2`
`__end2, _BinaryPredicate __pred)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 std::parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2`
`__end2, _BinaryPredicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _ForwardIterator, typename _Searcher >`
`_ForwardIterator std::parallel::search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &↵`
`__searcher)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator std::parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator std::parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val,`
`__gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator std::parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val,`
`_BinaryPredicate __binary_pred)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator std::parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val,`
`_BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_intersection (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_intersection (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 ↵`
`__end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_exact_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_sampling_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`

- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::__parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::__parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, __gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate`
`__pred)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate`
`__pred, __gnu_parallel::sequential_tag)`

7.588.1 Detailed Description

Parallel STL function calls corresponding to the `stl_algo.h` header.

The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

7.589 `algo.h` File Reference

Namespaces

- namespace `std`
- namespace `std::parallel`

Functions

- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool std::parallel::equal_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::equal_switch_tag, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::equal_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, __gnu_parallel::equal_switch_tag, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool std::parallel::lexicographical_compare_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair<_Iter1, _Iter2> std::parallel::mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair<_Iter1, _Iter2> std::parallel::mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair<_RAIter1, _RAIter2> std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair<_RAIter1, _RAIter2> std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`

- `template<typename _Iter1, typename _Iter2 >`
`bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`constexpr bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag, _BinaryPredicate __binary_pred)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`constexpr bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::__parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`constexpr bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`constexpr bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::__parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __begin1, _InputIterator1 __end1, _InputIterator2 __begin2, _InputIterator2 __end2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`

7.589.1 Detailed Description

Parallel STL function calls corresponding to the `stl_algobase.h` header. The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

7.590 `balanced_quicksort.h` File Reference

Classes

- struct [__gnu_parallel::__QSBThreadLocal<_RAIter>](#)

Namespaces

- namespace [__gnu_parallel](#)

Functions

- template<typename _RAIter, typename _Compare>
void [__gnu_parallel::__parallel_sort_qsb](#) (_RAIter __begin, _RAIter __end, _Compare __comp, [_ThreadIndex](#) __num_threads)
- template<typename _RAIter, typename _Compare>
void [__gnu_parallel::__qsb_conquer](#) ([_QSBThreadLocal](#)<_RAIter> **__tls, _RAIter __begin, _RAIter __end, _Compare __comp, [_ThreadIndex](#) __iam, [_ThreadIndex](#) __num_threads, bool __parent_wait)
- template<typename _RAIter, typename _Compare>
[std::iterator_traits](#)<_RAIter>::difference_type [__gnu_parallel::__qsb_divide](#) (_RAIter __begin, _RAIter __end, _Compare __comp, [_ThreadIndex](#) __num_threads)
- template<typename _RAIter, typename _Compare>
void [__gnu_parallel::__qsb_local_sort_with_helping](#) ([_QSBThreadLocal](#)<_RAIter> **__tls, _Compare &__comp, [_ThreadIndex](#) __iam, bool __wait)

7.590.1 Detailed Description

Implementation of a dynamically load-balanced parallel quicksort.

It works in-place and needs only logarithmic extra memory. The algorithm is similar to the one proposed in P. Tsigas and Y. Zhang. A simple, fast parallel implementation of quicksort and its performance evaluation on SUN enterprise 10000. In 11th Euromicro Conference on Parallel, Distributed and Network-Based Processing, page 372, 2003.

This file is a GNU parallel extension to the Standard C++ Library.

7.591 `base.h` File Reference

Classes

- class [__gnu_parallel::__binder1st<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType>](#)
- class [__gnu_parallel::__binder2nd<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType>](#)
- class [__gnu_parallel::__unary_negate<_Predicate, argument_type>](#)
- class [__gnu_parallel::__EqualFromLess<_T1, _T2, _Compare>](#)
- struct [__gnu_parallel::__EqualTo<_T1, _T2>](#)
- struct [__gnu_parallel::__Less<_T1, _T2>](#)
- struct [__gnu_parallel::__Multiplies<_Tp1, _Tp2, _Result>](#)
- struct [__gnu_parallel::__Plus<_Tp1, _Tp2, _Result>](#)
- class [__gnu_parallel::__PseudoSequence<_Tp, _DifferenceTp>](#)
- class [__gnu_parallel::__PseudoSequenceIterator<_Tp, _DifferenceTp>](#)

Namespaces

- namespace [__gnu_parallel](#)
- namespace [__gnu_sequential](#)
- namespace [std](#)
- namespace [std::__parallel](#)

Macros

- `#define _GLIBCXX_PARALLEL_ASSERT(_Condition)`

Functions

- `void __gnu_parallel::__decode2 (_CASable __x, int &__a, int &__b)`
- `_CASable __gnu_parallel::__encode2 (int __a, int __b)`
- `_ThreadIndex __gnu_parallel::__get_max_threads ()`
- `bool __gnu_parallel::__is_parallel (const _Parallelism __p)`
- `template<typename _RAIter, typename _Compare >
_RAIter __gnu_parallel::__median_of_three_iterators (_RAIter __a, _RAIter __b, _RAIter __c, _Compare __comp)`
- `template<typename _Size >
_Size __gnu_parallel::__rd_log2 (_Size __n)`
- `template<typename _Tp >
const _Tp & __gnu_parallel::__max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp >
const _Tp & __gnu_parallel::__min (const _Tp &__a, const _Tp &__b)`

7.591.1 Detailed Description

Sequential helper functions. This file is a GNU parallel extension to the Standard C++ Library.

7.592 `basic_iterator.h` File Reference**7.592.1 Detailed Description**

Includes the original header files concerned with iterators except for stream iterators. This file is a GNU parallel extension to the Standard C++ Library.

7.593 `checkers.h` File Reference**Namespaces**

- namespace `__gnu_parallel`

Functions

- `template<typename _Iter, typename _Compare >
bool __gnu_parallel::__is_sorted (_Iter __begin, _Iter __end, _Compare __comp)`

7.593.1 Detailed Description

Routines for checking the correctness of algorithm results. This file is a GNU parallel extension to the Standard C++ Library.

7.594 `compiletime_settings.h` File Reference**Macros**

- `#define _GLIBCXX_CALL(__n)`
- `#define _GLIBCXX_PARALLEL_ASSERTIONS`
- `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`
- `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`
- `#define _GLIBCXX_SCALE_DOWN_FPU`
- `#define _GLIBCXX_VERBOSE_LEVEL`

7.594.1 Detailed Description

Defines on options concerning debugging and performance, at compile-time. This file is a GNU parallel extension to the Standard C++ Library.

7.594.2 Macro Definition Documentation

`_GLIBCXX_CALL`

```
#define _GLIBCXX_CALL(  
    __n )
```

Macro to produce log message when entering a function.

Parameters

<code>__n</code>	Input size.
------------------	-------------

See also

`_GLIBCXX_VERBOSE_LEVEL`

`_GLIBCXX_PARALLEL_ASSERTIONS`

```
#define _GLIBCXX_PARALLEL_ASSERTIONS
```

Switch on many `_GLIBCXX_PARALLEL_ASSERTIONS` in parallel code. Should be switched on only locally.

`_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`

```
#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1
```

Switch on many `_GLIBCXX_PARALLEL_ASSERTIONS` in parallel code. Consider the size of the L1 cache for `gnu_parallel::__parallel_random_shuffle()`.

`_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`

```
#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB
```

Switch on many `_GLIBCXX_PARALLEL_ASSERTIONS` in parallel code. Consider the size of the TLB for `gnu_parallel::__parallel_random_shuffle()`.

`_GLIBCXX_SCALE_DOWN_FPU`

```
#define _GLIBCXX_SCALE_DOWN_FPU
```

Use floating-point scaling instead of modulo for mapping random numbers to a range. This can be faster on certain CPUs.

`_GLIBCXX_VERBOSE_LEVEL`

```
#define _GLIBCXX_VERBOSE_LEVEL
```

Determine verbosity level of the parallel mode. Level 1 prints a message each time a parallel-mode function is entered.

7.595 `equally_split.h` File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<typename _DifferenceType, typename _OutputIterator >`
`_OutputIterator __gnu_parallel::__equally_split` (`_DifferenceType __n`, `_ThreadIndex __num_threads`, `_OutputIterator __s`)
- `template<typename _DifferenceType >`
`_DifferenceType __gnu_parallel::__equally_split_point` (`_DifferenceType __n`, `_ThreadIndex __num_threads`, `_ThreadIndex __thread_no`)

7.595.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

7.596 features.h File Reference

Macros

- `#define _GLIBCXX_BAL_QUICKSORT`
- `#define _GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`
- `#define _GLIBCXX_FIND_EQUAL_SPLIT`
- `#define _GLIBCXX_FIND_GROWING_BLOCKS`
- `#define _GLIBCXX_MERGESORT`
- `#define _GLIBCXX_QUICKSORT`
- `#define _GLIBCXX_TREE_DYNAMIC_BALANCING`
- `#define _GLIBCXX_TREE_FULL_COPY`
- `#define _GLIBCXX_TREE_INITIAL_SPLITTING`

7.596.1 Detailed Description

Defines on whether to include algorithm variants.

Less variants reduce executable size and compile time. This file is a GNU parallel extension to the Standard C++ Library.

7.596.2 Macro Definition Documentation

`_GLIBCXX_BAL_QUICKSORT`

```
#define _GLIBCXX_BAL_QUICKSORT
```

Include parallel dynamically load-balanced quicksort.

See also

`__gnu_parallel::Settings::sort_algorithm`

`_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`

```
#define _GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS
```

Include the equal-sized blocks variant for `std::find`.

See also

`__gnu_parallel::Settings::find_algorithm`

_GLIBCXX_FIND_EQUAL_SPLIT

```
#define _GLIBCXX_FIND_EQUAL_SPLIT
```

Include the equal splitting variant for `std::find`.

See also

`__gnu_parallel::_Settings::find_algorithm`

_GLIBCXX_FIND_GROWING_BLOCKS

```
#define _GLIBCXX_FIND_GROWING_BLOCKS
```

Include the growing blocks variant for `std::find`.

See also

`__gnu_parallel::_Settings::find_algorithm`

_GLIBCXX_MERGESORT

```
#define _GLIBCXX_MERGESORT
```

Include parallel multi-way mergesort.

See also

`__gnu_parallel::_Settings::sort_algorithm`

_GLIBCXX_QUICKSORT

```
#define _GLIBCXX_QUICKSORT
```

Include parallel unbalanced quicksort.

See also

`__gnu_parallel::_Settings::sort_algorithm`

_GLIBCXX_TREE_DYNAMIC_BALANCING

```
#define _GLIBCXX_TREE_DYNAMIC_BALANCING
```

Include the dynamic balancing variant for `_Rb_tree::insert_unique(_Iter beg, _Iter __end)`.

See also

`__gnu_parallel::_Rb_tree`

_GLIBCXX_TREE_FULL_COPY

```
#define _GLIBCXX_TREE_FULL_COPY
```

In order to sort the input sequence of `_Rb_tree::insert_unique(_Iter beg, _Iter __end)` a full copy of the input elements is done.

See also

`__gnu_parallel::_Rb_tree`

`_GLIBCXX_TREE_INITIAL_SPLITTING`

```
#define _GLIBCXX_TREE_INITIAL_SPLITTING
```

Include the initial splitting variant for `_Rb_tree::insert_unique(_Iter beg, _Iter __end)`.

See also

`__gnu_parallel::_Rb_tree`

7.597 find.h File Reference

Namespaces

- namespace [`__gnu_parallel`](#)

Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
[`std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template`](#) (`_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector`)
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
[`std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template`](#) (`_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, constant_size_blocks_tag`)
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
[`std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template`](#) (`_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, equal_split_tag`)
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
[`std::pair< _RAIter1, _RAIter2 > __gnu_parallel::__find_template`](#) (`_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, growing_blocks_tag`)

7.597.1 Detailed Description

Parallel implementation base for `std::find()`, `std::equal()` and related functions. This file is a GNU parallel extension to the Standard C++ Library.

7.598 find_selectors.h File Reference

Classes

- struct [`__gnu_parallel::__adjacent_find_selector`](#)
- struct [`__gnu_parallel::__find_first_of_selector< _FIterator >`](#)
- struct [`__gnu_parallel::__find_if_selector`](#)
- struct [`__gnu_parallel::__generic_find_selector`](#)
- struct [`__gnu_parallel::__mismatch_selector`](#)

Namespaces

- namespace [`__gnu_parallel`](#)

7.598.1 Detailed Description

`_Function` objects representing different tasks to be plugged into the parallel find algorithm. This file is a GNU parallel extension to the Standard C++ Library.

7.599 for_each.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _Iter, typename _UserOp, typename _Functionality, typename _Red, typename _Result >
_UserOp __gnu_parallel::__for_each_template_random_access (_Iter __begin, _Iter __end, _UserOp __user←
_op, _Functionality & __functionality, _Red __reduction, _Result __reduction_start, _Result & __output, typename
std::iterator_traits< _Iter >::difference_type __bound, _Parallelism __parallelism_tag)`

7.599.1 Detailed Description

Main interface for embarrassingly parallel functions.

The explicit implementation are in other header files, like `workstealing.h`, `par_loop.h`, `omp_loop.h`, and `omp_loop_←
static.h`. This file is a GNU parallel extension to the Standard C++ Library.

7.600 for_each_selectors.h File Reference

Classes

- struct [__gnu_parallel::__accumulate_binop_reduct](#)< _BinOp >
- struct [__gnu_parallel::__accumulate_selector](#)< _It >
- struct [__gnu_parallel::__adjacent_difference_selector](#)< _It >
- struct [__gnu_parallel::__count_if_selector](#)< _It, _Diff >
- struct [__gnu_parallel::__count_selector](#)< _It, _Diff >
- struct [__gnu_parallel::__fill_selector](#)< _It >
- struct [__gnu_parallel::__for_each_selector](#)< _It >
- struct [__gnu_parallel::__generate_selector](#)< _It >
- struct [__gnu_parallel::__generic_for_each_selector](#)< _It >
- struct [__gnu_parallel::__identity_selector](#)< _It >
- struct [__gnu_parallel::__inner_product_selector](#)< _It, _It2, _Tp >
- struct [__gnu_parallel::__max_element_reduct](#)< _Compare, _It >
- struct [__gnu_parallel::__min_element_reduct](#)< _Compare, _It >
- struct [__gnu_parallel::__replace_if_selector](#)< _It, _Op, _Tp >
- struct [__gnu_parallel::__replace_selector](#)< _It, _Tp >
- struct [__gnu_parallel::__transform1_selector](#)< _It >
- struct [__gnu_parallel::__transform2_selector](#)< _It >
- struct [__gnu_parallel::__DummyReduct](#)
- struct [__gnu_parallel::__Nothing](#)

Namespaces

- namespace [__gnu_parallel](#)

7.600.1 Detailed Description

Functors representing different tasks to be plugged into the generic parallelization methods for embarrassingly parallel functions. This file is a GNU parallel extension to the Standard C++ Library.

7.601 iterator.h File Reference

Classes

- class [__gnu_parallel::_IteratorPair<_Iterator1, _Iterator2, _IteratorCategory>](#)
- class [__gnu_parallel::_IteratorTriple<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory>](#)

Namespaces

- namespace [__gnu_parallel](#)

7.601.1 Detailed Description

Helper iterator classes for the `std::transform()` functions. This file is a GNU parallel extension to the Standard C++ Library.

7.602 list_partition.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _Iter>`
`void __gnu_parallel::_shrink (std::vector<_Iter> &__os_starts, size_t &__count_to_two, size_t &__range_↵
length)`
- `template<typename _Iter>`
`void __gnu_parallel::_shrink_and_double (std::vector<_Iter> &__os_starts, size_t &__count_to_two, size_t
&__range_length, const bool __make_twice)`
- `template<typename _Iter, typename _FuncType>`
`size_t __gnu_parallel::list_partition (const _Iter __begin, const _Iter __end, _Iter *__starts, size_t *__lengths,
const int __num_parts, _FuncType &__f, int __oversampling=0)`

7.602.1 Detailed Description

__Functionality to split __sequence referenced by only input iterators. This file is a GNU parallel extension to the Standard C++ Library.

7.603 losertree.h File Reference

Classes

- struct [__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser](#)
- struct [__gnu_parallel::_LoserTreePointerBase<_Tp, _Compare>::_Loser](#)
- class [__gnu_parallel::_LoserTree<__stable, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTree<false, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreeBase<_Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointer<__stable, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointer<false, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointerBase<_Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointerUnguarded<__stable, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointerUnguarded<false, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreePointerUnguardedBase<_Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreeUnguarded<__stable, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreeUnguarded<false, _Tp, _Compare>](#)
- class [__gnu_parallel::_LoserTreeUnguardedBase<_Tp, _Compare>](#)

Namespaces

- namespace [__gnu_parallel](#)

7.603.1 Detailed Description

Many generic loser tree variants. This file is a GNU parallel extension to the Standard C++ Library.

7.604 merge.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_movc (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_usual (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter3, typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter1 &__begin2, _RAIter1 __end2, _RAIter3 __target, typename std::iterator_traits<_RAIter1 >::difference_type __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _RAIter3 __target, typename std::iterator_traits<_RAIter1 >::difference_type __max_length, _Compare __comp)`

7.604.1 Detailed Description

Parallel implementation of `std::merge()`. This file is a GNU parallel extension to the Standard C++ Library.

7.605 multiseq_selection.h File Reference

Classes

- class [__gnu_parallel::_Lexicographic](#)<_T1, _T2, _Compare >
- class [__gnu_parallel::_LexicographicReverse](#)<_T1, _T2, _Compare >

Namespaces

- namespace [__gnu_parallel](#)

Macros

- `#define __S(__i)`
- `#define __S(__i)`

Functions

- `template<typename _RanSeqs, typename _RankType, typename _RankIterator, typename _Compare >`
`void __gnu_parallel::multiseq_partition (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankIterator __begin_offsets, _Compare __comp=std::less< typename std::iterator_traits< typename _RanSeqs >::value_type::first_type >::value_type >())`
- `template<typename _Tp, typename _RanSeqs, typename _RankType, typename _Compare >`
`_Tp __gnu_parallel::multiseq_selection (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankType __offset, _Compare __comp=std::less< _Tp >())`

7.605.1 Detailed Description

Functions to find elements of a certain global __rank in multiple sorted sequences. Also serves for splitting such sequence sets.

The algorithm description can be found in

P. J. Varman, S. D. Scheufler, B. R. Iyer, and G. R. Ricard. Merging Multiple Lists on Hierarchical-Memory Multiprocessors. Journal of Parallel and Distributed Computing, 12(2):171-177, 1991.

This file is a GNU parallel extension to the Standard C++ Library.

7.606 multiway_merge.h File Reference

Classes

- `struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >`
- `class __gnu_parallel::_GuardedIterator< _RAIter, _Compare >`
- `struct __gnu_parallel::_LoserTreeTraits< _Tp >`
- `struct __gnu_parallel::_SamplingSorter< __stable, _RAIter, _StrictWeakOrdering >`
- `struct __gnu_parallel::_SamplingSorter< false, _RAIter, _StrictWeakOrdering >`

Namespaces

- namespace `__gnu_parallel`

Macros

- `#define _GLIBCXX_PARALLEL_DECISION(__a, __b, __c, __d)`
- `#define _GLIBCXX_PARALLEL_LENGTH(__s)`
- `#define _GLIBCXX_PARALLEL_MERGE_3_CASE(__a, __b, __c, __c0, __c1)`
- `#define _GLIBCXX_PARALLEL_MERGE_4_CASE(__a, __b, __c, __d, __c0, __c1, __c2)`

Functions

- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __gnu_parallel::merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<bool __stable, bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 __gnu_parallel::sequential_multiway_merge (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`

- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sampling_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
_RAIter3 gnu_parallel::multiway_merge_3_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<template< typename _RAI, typename _Cp > class iterator, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
_RAIter3 gnu_parallel::multiway_merge_4_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >
void gnu_parallel::multiway_merge_exact_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, _DifferenceType > > *__pieces)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
_RAIter3 gnu_parallel::multiway_merge_loser_tree (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<typename _UnguardedLoserTree, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
_RAIter3 gnu_parallel::multiway_merge_loser_tree_sentinel (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
_RAIter3 gnu_parallel::multiway_merge_loser_tree_unguarded (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >
void gnu_parallel::multiway_merge_sampling_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, _DifferenceType > > *__pieces)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`

- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<bool __stable, bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Splitter, typename _Compare >`
`_RAIter3 gnu_parallel::parallel_multiway_merge (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _Splitter __splitter, _DifferenceTp __length, _Compare __comp, ThreadIndex __num_threads)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`

7.606.1 Detailed Description

Implementation of sequential and parallel multiway merge.

Explanations on the high-speed merging routines in the appendix of

P. Sanders. Fast priority queues for cached memory. ACM Journal of Experimental Algorithmics, 5, 2000.

This file is a GNU parallel extension to the Standard C++ Library.

7.606.2 Macro Definition Documentation

`_GLIBCXX_PARALLEL_LENGTH`

```
#define _GLIBCXX_PARALLEL_LENGTH(
    __s )
```

Length of a sequence described by a pair of iterators.

7.607 `multiway_mergesort.h` File Reference

Classes

- struct `__gnu_parallel::Piece<_DifferenceTp>`
- struct `__gnu_parallel::PMWMSortingData<_RAIter>`
- struct `__gnu_parallel::SplitConsistently<__exact, _RAIter, _Compare, _SortingPlacesIterator>`
- struct `__gnu_parallel::SplitConsistently<false, _RAIter, _Compare, _SortingPlacesIterator>`
- struct `__gnu_parallel::SplitConsistently<true, _RAIter, _Compare, _SortingPlacesIterator>`

Namespaces

- namespace `__gnu_parallel`

Functions

- template<typename `_RAIter`, typename `_DifferenceTp`>
void `__gnu_parallel::__determine_samples` (`PMWMSortingData<_RAIter> *__sd`, `_DifferenceTp __num`↵
`__samples`)
- template<bool `__stable`, bool `__exact`, typename `_RAIter`, typename `_Compare`>
void `__gnu_parallel::parallel_sort_mwms` (`_RAIter __begin`, `_RAIter __end`, `_Compare __comp`, `ThreadIndex`
`__num_threads`)
- template<bool `__stable`, bool `__exact`, typename `_RAIter`, typename `_Compare`>
void `__gnu_parallel::parallel_sort_mwms_pu` (`PMWMSortingData<_RAIter> *__sd`, `_Compare &__comp`)

7.607.1 Detailed Description

Parallel multiway merge sort. This file is a GNU parallel extension to the Standard C++ Library.

7.608 `numericfwd.h` File Reference

Namespaces

- namespace `std`
- namespace `std::__parallel`

Functions

- template<typename `_Iter`, typename `_Tp`, typename `_BinaryOper`, typename `_Tag`>
`_Tp std::__parallel::__accumulate_switch` (`_Iter`, `_Iter`, `_Tp`, `_BinaryOper`, `_Tag`)
- template<typename `_Iter`, typename `_Tp`, typename `_Tag`>
`_Tp std::__parallel::__accumulate_switch` (`_Iter`, `_Iter`, `_Tp`, `_Tag`)
- template<typename `_RAIter`, typename `_Tp`, typename `_BinaryOper`>
`_Tp std::__parallel::__accumulate_switch` (`_RAIter`, `_RAIter`, `_Tp`, `_BinaryOper`, `random_access_iterator_tag`,
`__gnu_parallel::Parallelism __parallelism=__gnu_parallel::parallel_unbalanced`)
- template<typename `_Iter`, typename `_OIter`, typename `_BinaryOper`, typename `_Tag1`, typename `_Tag2`>
`_OIter std::__parallel::__adjacent_difference_switch` (`_Iter`, `_Iter`, `_OIter`, `_BinaryOper`, `_Tag1`, `_Tag2`)

- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::__adjacent_difference_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag,
random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename`
`_Tag1, typename _Tag2 >`
`_Tp std::__parallel::__inner_product_switch (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _Binary↵`
`Function2, _Tag1, _Tag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::__inner_product_switch (_RAIter1, _RAIter1, _RAIter2, _Tp, _BinaryFunction1, _Binary↵`
`Function2, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::__Parallelism=__gnu_parallel::parallel_unbal`
- `template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >`
`_OIter std::__parallel::__partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::__partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag,
random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, _BinaryOper)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::__adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::__Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2,
__gnu_parallel::__Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::__inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2,
__gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter __result)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`

7.608.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

7.609 omp_loop.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __gnu_parallel::__for_each_template_random_access_omp_loop (_RAIter __begin, _RAIter __end, _Op ↵
_o, _Fu &_f, _Red _r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >↵
::difference_type __bound)`

7.609.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of an OpenMP for loop. This file is a GNU parallel extension to the Standard C++ Library.

7.610 omp_loop_static.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __gnu_parallel::__for_each_template_random_access_omp_loop_static (_RAIter __begin, _RAIter __end,
_Op __o, _Fu &_f, _Red _r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >↵
::difference_type __bound)`

7.610.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of an OpenMP for loop with static scheduling. This file is a GNU parallel extension to the Standard C++ Library.

7.611 par_loop.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __gnu_parallel::__for_each_template_random_access_ed` (`_RAIter __begin`, `_RAIter __end`, `_Op __o`, `_Fu`
`& __f`, `_Red __r`, `_Result __base`, `_Result & __output`, `typename std::iterator_traits<_RAIter>::difference_type`
`__bound`)

7.611.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of equal splitting. This file is a GNU parallel extension to the Standard C++ Library.

7.612 parallel.h File Reference

7.612.1 Detailed Description

End-user include file. Provides advanced settings and tuning options. This file is a GNU parallel extension to the Standard C++ Library.

7.613 partial_sum.h File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum` (`_Iter __begin`, `_Iter __end`, `_OutputIterator __result`, `↔`
`_BinaryOperation __bin_op`)
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum_basecase` (`_Iter __begin`, `_Iter __end`, `_OutputIterator`
`__result`, `_BinaryOperation __bin_op`, `typename std::iterator_traits<_Iter>::value_type __value`)
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum_linear` (`_Iter __begin`, `_Iter __end`, `_OutputIterator __↔`
`result`, `_BinaryOperation __bin_op`, `typename std::iterator_traits<_Iter>::difference_type __n`)

7.613.1 Detailed Description

Parallel implementation of `std::partial_sum()`, i.e. prefix sums. This file is a GNU parallel extension to the Standard C++ Library.

7.614 partition.h File Reference

Namespaces

- namespace `__gnu_parallel`

Macros

- `#define _GLIBCXX_VOLATILE`

Functions

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_nth_element` (`_RAIter __begin`, `_RAIter __nth`, `_RAIter __end`, `_Compare __↔`
`comp`)

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __↵
comp)`
- `template<typename _RAIter, typename _Predicate >`
`std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__parallel_partition (_RAIter __begin, _RAIter ↵
__end, _Predicate __pred, _ThreadIndex __num_threads)`

7.614.1 Detailed Description

Parallel implementation of `std::partition()`, `std::nth_element()`, and `std::partial_sort()`. This file is a GNU parallel extension to the Standard C++ Library.

7.614.2 Macro Definition Documentation

`_GLIBCXX_VOLATILE`

```
#define _GLIBCXX_VOLATILE
```

Decide whether to declare certain variables volatile.

7.615 `queue.h` File Reference

Classes

- class [__gnu_parallel::_RestrictedBoundedConcurrentQueue](#)< _Tp >

Namespaces

- namespace [__gnu_parallel](#)

Macros

- `#define _GLIBCXX_VOLATILE`

7.615.1 Detailed Description

Lock-free double-ended queue. This file is a GNU parallel extension to the Standard C++ Library.

7.615.2 Macro Definition Documentation

`_GLIBCXX_VOLATILE`

```
#define _GLIBCXX_VOLATILE
```

Decide whether to declare certain variable volatile in this file.

7.616 `quicksort.h` File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort_qs (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex ↵
__num_threads)`

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort_qs_conquer (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__parallel_sort_qs_divide (_RAIter __begin, _RAIter __end, _Compare __comp, typename std::iterator_traits< _RAIter >::difference_type __pivot_rank, typename std::iterator_traits< _RAIter >::difference_type __num_samples, _ThreadIndex __num_threads)`

7.616.1 Detailed Description

Implementation of a unbalanced parallel quicksort (in-place). This file is a GNU parallel extension to the Standard C++ Library.

7.617 random_number.h File Reference

Classes

- class `__gnu_parallel::_RandomNumber`

Namespaces

- namespace `__gnu_parallel`

7.617.1 Detailed Description

Random number generator based on the Mersenne twister. This file is a GNU parallel extension to the Standard C++ Library.

7.618 random_shuffle.h File Reference

Classes

- struct `__gnu_parallel::_DRandomShufflingGlobalData< _RAIter >`
- struct `__gnu_parallel::_DRSSorterPU< _RAIter, _RandomNumberGenerator >`

Namespaces

- namespace `__gnu_parallel`

Typedefs

- typedef unsigned short `__gnu_parallel::_BinIndex`

Functions

- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __gnu_parallel::__parallel_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator __rng= _RandomNumber())`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __gnu_parallel::__parallel_random_shuffle_drs (_RAIter __begin, _RAIter __end, typename std::iterator_traits< _RAIter >::difference_type __n, _ThreadIndex __num_threads, _RandomNumberGenerator &__rng)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __gnu_parallel::__parallel_random_shuffle_drs_pu (_DRSSorterPU< _RAIter, _RandomNumberGenerator > *__pus)`
- `template<typename _RandomNumberGenerator >`
`int __gnu_parallel::__random_number_pow2 (int __logp, _RandomNumberGenerator &__rng)`

- `template<typename _Tp >`
`_Tp __gnu_parallel::__round_up_to_pow2 (_Tp __x)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __gnu_parallel::__sequential_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &__rng)`

7.618.1 Detailed Description

Parallel implementation of `std::random_shuffle()`. This file is a GNU parallel extension to the Standard C++ Library.

7.619 search.h File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<typename _RAIter, typename _DifferenceTp >`
`void __gnu_parallel::__calc_borders (_RAIter __elements, _DifferenceTp __length, _DifferenceTp *__off)`
- `template<typename __RAIter1, typename __RAIter2, typename _Pred >`
`__RAIter1 __gnu_parallel::__search_template (__RAIter1 __begin1, __RAIter1 __end1, __RAIter2 __begin2, __RAIter2 __end2, _Pred __pred)`

7.619.1 Detailed Description

Parallel implementation base for `std::search()` and `std::search_n()`. This file is a GNU parallel extension to the Standard C++ Library.

7.620 set_operations.h File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator __gnu_parallel::__copy_tail (std::pair< _Iter, _Iter > __b, std::pair< _Iter, _Iter > __e, _OutputIterator __r)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_intersection (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Operation >`
`_OutputIterator __gnu_parallel::__parallel_set_operation (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Operation __op)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_symmetric_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_union (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`

7.620.1 Detailed Description

Parallel implementations of set operations for random-access iterators. This file is a GNU parallel extension to the Standard C++ Library.

7.621 settings.h File Reference

Classes

- struct `__gnu_parallel::_Settings`

Namespaces

- namespace `__gnu_parallel`

Macros

- `#define _GLIBCXX_PARALLEL_CONDITION(__c)`

7.621.1 Detailed Description

Runtime settings and tuning parameters, heuristics to decide whether to use parallelized algorithms. This file is a GNU parallel extension to the Standard C++ Library.

7.621.2 Deciding whether to run an algorithm in parallel.

There are several ways the user can switch on and off the parallel execution of an algorithm, both at compile- and run-time.

Only sequential execution can be forced at compile-time. This reduces code size and protects code parts that have non-thread-safe side effects.

Ultimately, forcing parallel execution at compile-time makes sense. Often, the sequential algorithm implementation is used as a subroutine, so no reduction in code size can be achieved. Also, the machine the program is run on might have only one processor core, so to avoid overhead, the algorithm is executed sequentially.

To force sequential execution of an algorithm ultimately at compile-time, the user must add the tag `gnu_parallel::sequential_tag()` to the end of the parameter list, e. g.

```
std::sort(__v.begin(), __v.end(), __gnu_parallel::sequential_tag());
```

This is compatible with all overloaded algorithm variants. No additional code will be instantiated, at all. The same holds for most algorithm calls with iterators not providing random access.

If the algorithm call is not forced to be executed sequentially at compile-time, the decision is made at run-time. The global variable `__gnu_parallel::_Settings::algorithm_strategy` is checked. It is a tristate variable corresponding to:

- a. `force_sequential`, meaning the sequential algorithm is executed.
- b. `force_parallel`, meaning the parallel algorithm is executed.
- c. `heuristic`

For heuristic, the parallel algorithm implementation is called only if the input size is sufficiently large. For most algorithms, the input size is the (combined) length of the input sequence(`__s`). The threshold can be set by the user, individually for each algorithm. The according variables are called `gnu_parallel::_Settings::[algorithm]_minimal_n`.

For some of the algorithms, there are even more tuning options, e. g. the ability to choose from multiple algorithm variants. See below for details.

7.621.3 Macro Definition Documentation

`_GLIBCXX_PARALLEL_CONDITION`

```
#define _GLIBCXX_PARALLEL_CONDITION(  
    __c )
```

Determine at compile(?)-time if the parallel variant of an algorithm should be called.

Parameters

<code>__gnu_parallel::__c</code>	A condition that is convertible to bool that is overruled by <code>__gnu_parallel::Settings::algorithm_strategy</code> . Usually a decision based on the input size.
----------------------------------	--

7.622 sort.h File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<bool __stable, typename _RAIter, typename _Compare, typename _Parallelism >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, balanced_quicksort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, default_parallel_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_exact_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_sampling_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, parallel_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, quicksort_tag __parallelism)`

7.622.1 Detailed Description

Parallel sorting algorithm switch. This file is a GNU parallel extension to the Standard C++ Library.

7.623 tags.h File Reference

Classes

- struct `__gnu_parallel::balanced_quicksort_tag`
- struct `__gnu_parallel::balanced_tag`
- struct `__gnu_parallel::constant_size_blocks_tag`
- struct `__gnu_parallel::default_parallel_tag`
- struct `__gnu_parallel::equal_split_tag`

- struct [__gnu_parallel::exact_tag](#)
- struct [__gnu_parallel::find_tag](#)
- struct [__gnu_parallel::growing_blocks_tag](#)
- struct [__gnu_parallel::multiway_mergesort_exact_tag](#)
- struct [__gnu_parallel::multiway_mergesort_sampling_tag](#)
- struct [__gnu_parallel::multiway_mergesort_tag](#)
- struct [__gnu_parallel::omp_loop_static_tag](#)
- struct [__gnu_parallel::omp_loop_tag](#)
- struct [__gnu_parallel::parallel_tag](#)
- struct [__gnu_parallel::quicksort_tag](#)
- struct [__gnu_parallel::sampling_tag](#)
- struct [__gnu_parallel::sequential_tag](#)
- struct [__gnu_parallel::unbalanced_tag](#)

Namespaces

- namespace [__gnu_parallel](#)

7.623.1 Detailed Description

Tags for compile-time selection. This file is a GNU parallel extension to the Standard C++ Library.

7.624 types.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Typedefs

- typedef int64_t [__gnu_parallel::_CASable](#)
- typedef uint64_t [__gnu_parallel::_SequenceIndex](#)
- typedef uint16_t [__gnu_parallel::_ThreadIndex](#)

Enumerations

- enum [__gnu_parallel::_AlgorithmStrategy](#) { **heuristic** , **force_sequential** , **force_parallel** }
- enum [__gnu_parallel::_FindAlgorithm](#) { **GROWING_BLOCKS** , **CONSTANT_SIZE_BLOCKS** , **EQUAL_SPLIT** }
- enum [__gnu_parallel::_MultiwayMergeAlgorithm](#) { **LOSER_TREE** }
- enum [__gnu_parallel::_Parallelism](#) { [__gnu_parallel::sequential](#) , [__gnu_parallel::parallel_unbalanced](#) , [__gnu_parallel::parallel_balanced](#) , [__gnu_parallel::parallel_omp_loop](#) , [__gnu_parallel::parallel_omp_loop_static](#) , [__gnu_parallel::parallel_taskqueue](#) }
- enum [__gnu_parallel::_PartialSumAlgorithm](#) { **RECURSIVE** , **LINEAR** }
- enum [__gnu_parallel::_SortAlgorithm](#) { **MWMS** , **QS** , **QS_BALANCED** }
- enum [__gnu_parallel::_SplittingAlgorithm](#) { **SAMPLING** , **EXACT** }

Variables

- static const int [__gnu_parallel::_CASable_bits](#)
- static const [_CASable](#) [__gnu_parallel::_CASable_mask](#)

7.624.1 Detailed Description

Basic types and typedefs. This file is a GNU parallel extension to the Standard C++ Library.

7.625 `unique_copy.h` File Reference

Namespaces

- namespace `__gnu_parallel`

Functions

- `template<typename _Iter, class _OutputIterator >
_OutputIterator __gnu_parallel::__parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result)`
- `template<typename _Iter, class _OutputIterator, class _BinaryPredicate >
_OutputIterator __gnu_parallel::__parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result, ↔
BinaryPredicate __binary_pred)`

7.625.1 Detailed Description

Parallel implementations of `std::unique_copy()`. This file is a GNU parallel extension to the Standard C++ Library.

7.626 `workstealing.h` File Reference

Classes

- struct `__gnu_parallel::__Job<_DifferenceTp >`

Namespaces

- namespace `__gnu_parallel`

Macros

- `#define _GLIBCXX_JOB_VOLATILE`

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_workstealing (_RAIter __begin, _RAIter __end, ↔
Op __op, _Fu &__f, _Red __r, _Result __base, _Result &__output, typename std::iterator_traits<_RAIter >↔
::difference_type __bound)`

7.626.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of work-stealing.

Work stealing is described in

R. D. Blumofe and C. E. Leiserson. Scheduling multithreaded computations by work stealing. *Journal of the ACM*, 46(5):720-748, 1999.

This file is a GNU parallel extension to the Standard C++ Library.

7.627 `queue` File Reference

Macros

- `#define _GLIBCXX_QUEUE`

7.627.1 Detailed Description

This is a Standard C++ Library header.

7.628 random File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_randint`
- `#define _GLIBCXX_EXPERIMENTAL_RANDOM`

Functions

- [std::default_random_engine](#) & [std::experimental::_S_randint_engine](#) ()
- `template<typename _IntType >`
`_IntType std::experimental::randint (_IntType __a, _IntType __b)`
- `void std::experimental::reseed ()`
- `void std::experimental::reseed (default_random_engine::result_type __value)`

7.628.1 Detailed Description

This is a TS C++ Library header.

7.629 random File Reference

Macros

- `#define _GLIBCXX_RANDOM`

7.629.1 Detailed Description

This is a Standard C++ Library header.

7.630 ranges File Reference

Classes

- class [std::ranges::empty_view< _Tp >](#)
- class [std::ranges::single_view< _Tp >](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_RANGES`

Typedefs

- `template<typename _Wp >`
`using std::ranges::__detail::__iota_diff_t = decltype(__to_signed_like(std::declval< _Wp >()))`
- `template<bool _Const, typename _Tp >`
`using std::ranges::__detail::__maybe_const_t = __conditional_t< _Const, const _Tp, _Tp >`

- `template<bool _Present, typename _Tp >`
`using std::ranges::__detail::__maybe_present_t = __conditional_t< _Present, _Tp, _Empty >`
- `template<viewable_range _Range>`
`using std::ranges::views::all_t = decltype(all(std::declval< _Range >()))`
- `template<typename _Val >`
`using std::ranges::istream_view = basic_istream_view< _Val, char >`
- `template<typename _Range >`
`using std::ranges::keys_view = elements_view< views::all_t< _Range >, 0 >`
- `template<typename _Range >`
`using std::ranges::values_view = elements_view< views::all_t< _Range >, 1 >`
- `template<typename _Val >`
`using std::ranges::wistream_view = basic_istream_view< _Val, wchar_t >`

Functions

- `template<typename _Wp >`
`constexpr auto std::ranges::__detail::__to_signed_like (_Wp __w) noexcept`
- `template<typename _Range >`
`std::ranges::common_view (_Range &&) -> common_view< views::all_t< _Range > >`
- `template<typename _Range >`
`std::ranges::drop_view (_Range &&, range_difference_t< _Range >) -> drop_view< views::all_t< _Range > >`
- `template<typename _Range, typename _Pred >`
`std::ranges::drop_while_view (_Range &&, _Pred) -> drop_while_view< views::all_t< _Range >, _Pred >`
- `template<typename _Range, typename _Pred >`
`std::ranges::filter_view (_Range &&, _Pred) -> filter_view< views::all_t< _Range >, _Pred >`
- `template<typename _Winc, typename _Bound >`
`requires (!__detail::__is_integer_like<_Winc> || !__detail::__is_integer_like<_Bound> || (__detail::__is_signed_integer_like<_Winc> == __detail::__is_signed_integer_like<_Bound>))`
`std::ranges::iota_view (_Winc, _Bound) -> iota_view< _Winc, _Bound >`
- `template<typename _Range >`
`std::ranges::join_view (_Range &&) -> join_view< views::all_t< _Range > >`
- `template<typename _Range, typename _Pattern >`
`std::ranges::lazy_split_view (_Range &&, _Pattern &&) -> lazy_split_view< views::all_t< _Range >, views::all_t< _Pattern > >`
- `template<input_range _Range>`
`std::ranges::lazy_split_view (_Range &&, range_value_t< _Range >) -> lazy_split_view< views::all_t< _Range >, single_view< range_value_t< _Range > > >`
- `template<typename _Range >`
`std::ranges::ref_view (_Range &) -> ref_view< _Range >`
- `template<typename _Range >`
`std::ranges::reverse_view (_Range &&) -> reverse_view< views::all_t< _Range > >`
- `template<typename _Tp >`
`std::ranges::single_view (_Tp) -> single_view< _Tp >`
- `template<typename _Range, typename _Pattern >`
`std::ranges::split_view (_Range &&, _Pattern &&) -> split_view< views::all_t< _Range >, views::all_t< _Pattern > >`
- `template<forward_range _Range>`
`std::ranges::split_view (_Range &&, range_value_t< _Range >) -> split_view< views::all_t< _Range >, single_view< range_value_t< _Range > > >`
- `template<typename _Range >`
`std::ranges::take_view (_Range &&, range_difference_t< _Range >) -> take_view< views::all_t< _Range > >`

- `template<typename _Range, typename _Pred >`
`std::ranges::take_while_view` (`_Range &&`, `_Pred`) -> `take_while_view< views::all_t< _Range >, _Pred >`
- `template<typename _Range, typename _Fp >`
`std::ranges::transform_view` (`_Range &&`, `_Fp`) -> `transform_view< views::all_t< _Range >, _Fp >`

Variables

- `template<typename _Range >`
`constexpr bool std::ranges::views::__detail::__is_basic_string_view`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::ranges::views::__detail::__is_basic_string_view< basic_string_view< _CharT, _Traits > >`
- `template<typename _Range >`
`constexpr bool std::ranges::views::__detail::__is_empty_view`
- `template<typename _Tp >`
`constexpr bool std::ranges::views::__detail::__is_empty_view< empty_view< _Tp > >`
- `template<typename _Range >`
`constexpr bool std::ranges::views::__detail::__is_iota_view`
- `template<typename _Winc, typename _Bound >`
`constexpr bool std::ranges::views::__detail::__is_iota_view< iota_view< _Winc, _Bound > >`
- `template<typename >`
`constexpr bool std::ranges::views::__detail::__is_reverse_view`
- `template<typename _Vp >`
`constexpr bool std::ranges::views::__detail::__is_reverse_view< reverse_view< _Vp > >`
- `template<typename >`
`constexpr bool std::ranges::views::__detail::__is_reversible_subrange`
- `template<typename _Iter, subrange_kind _Kind >`
`constexpr bool std::ranges::views::__detail::__is_reversible_subrange< subrange< reverse_iterator< ↵
_Iter >, reverse_iterator< _Iter >, _Kind > >`
- `template<typename _Range >`
`constexpr bool std::ranges::views::__detail::__is_subrange`
- `template<typename _Iter, typename _Sent, subrange_kind _Kind >`
`constexpr bool std::ranges::views::__detail::__is_subrange< subrange< _Iter, _Sent, _Kind > >`
- `constexpr _All std::ranges::views::all`
- `constexpr _Common std::ranges::views::common`
- `constexpr _Counted std::ranges::views::counted`
- `constexpr _Drop std::ranges::views::drop`
- `constexpr _DropWhile std::ranges::views::drop_while`
- `template<size_t _Nm >`
`constexpr _Elements< _Nm > std::ranges::views::elements`
- `template<typename _Tp >`
`constexpr empty_view< _Tp > std::ranges::views::empty`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< common_view< _Tp > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< drop_view< _Tp > >`
- `template<typename _Tp, typename _Pred >`
`constexpr bool std::ranges::enable_borrowed_range< drop_while_view< _Tp, _Pred > >`
- `template<typename _Tp, size_t _Nm >`
`constexpr bool std::ranges::enable_borrowed_range< elements_view< _Tp, _Nm > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< empty_view< _Tp > >`

- `template<typename _Winc, typename _Bound >`
`constexpr bool std::ranges::enable_borrowed_range< iota_view< _Winc, _Bound > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< owning_view< _Tp > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< ref_view< _Tp > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< reverse_view< _Tp > >`
- `template<typename _Tp >`
`constexpr bool std::ranges::enable_borrowed_range< take_view< _Tp > >`
- `constexpr _Filter std::ranges::views::filter`
- `constexpr _Iota std::ranges::views::iota`
- `template<typename _Tp >`
`constexpr _Istream< _Tp > std::ranges::views::istream`
- `constexpr _Join std::ranges::views::join`
- `constexpr auto std::ranges::views::keys`
- `constexpr _LazySplit std::ranges::views::lazy_split`
- `constexpr _Reverse std::ranges::views::reverse`
- `constexpr _Single std::ranges::views::single`
- `constexpr _Split std::ranges::views::split`
- `constexpr _Take std::ranges::views::take`
- `constexpr _TakeWhile std::ranges::views::take_while`
- `constexpr _Transform std::ranges::views::transform`
- `constexpr auto std::ranges::views::values`

7.630.1 Detailed Description

This is a Standard C++ Library header.

7.631 ratio File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_RATIO`

Variables

- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_greater_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_greater_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_less_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_less_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::ratio_not_equal_v`

7.631.1 Detailed Description

This is a TS C++ Library header.

7.632 ratio File Reference

Classes

- struct [std::ratio< _Num, _Den >](#)
- struct [std::ratio_equal< _R1, _R2 >](#)
- struct [std::ratio_greater< _R1, _R2 >](#)
- struct [std::ratio_greater_equal< _R1, _R2 >](#)
- struct [std::ratio_less< _R1, _R2 >](#)
- struct [std::ratio_less_equal< _R1, _R2 >](#)
- struct [std::ratio_not_equal< _R1, _R2 >](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_RATIO`

Typedefs

- typedef [ratio< 1, 1000000000000000000 >](#) **std::atto**
- typedef [ratio< 1, 100 >](#) **std::centi**
- typedef [ratio< 10, 1 >](#) **std::deca**
- typedef [ratio< 1, 10 >](#) **std::deci**
- typedef [ratio< 1000000000000000000, 1 >](#) **std::exa**
- typedef [ratio< 1, 10000000000000000 >](#) **std::femto**
- typedef [ratio< 1000000000, 1 >](#) **std::giga**
- typedef [ratio< 100, 1 >](#) **std::hecto**
- typedef [ratio< 1000, 1 >](#) **std::kilo**
- typedef [ratio< 1000000, 1 >](#) **std::mega**
- typedef [ratio< 1, 1000000 >](#) **std::micro**
- typedef [ratio< 1, 1000 >](#) **std::milli**
- typedef [ratio< 1, 1000000000 >](#) **std::nano**
- typedef [ratio< 1000000000000000000, 1 >](#) **std::peta**
- typedef [ratio< 1, 1000000000000 >](#) **std::pico**
- template<typename _R1, typename _R2 >
using [std::ratio_add](#) = typename __ratio_add< _R1, _R2 >::type
- template<typename _R1, typename _R2 >
using [std::ratio_divide](#) = typename __ratio_divide< _R1, _R2 >::type
- template<typename _R1, typename _R2 >
using [std::ratio_multiply](#) = typename __ratio_multiply< _R1, _R2 >::type
- template<typename _R1, typename _R2 >
using [std::ratio_subtract](#) = typename __ratio_subtract< _R1, _R2 >::type
- typedef [ratio< 1000000000000, 1 >](#) **std::tera**

Variables

- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_equal_v`
- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_greater_equal_v`
- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_greater_v`
- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_less_equal_v`
- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_less_v`
- `template<typename _R1 , typename _R2 >`
`constexpr bool std::ratio_not_equal_v`

7.632.1 Detailed Description

This is a Standard C++ Library header.

7.633 ratio File Reference

Namespaces

- namespace [std](#)
- namespace [std::tr2](#)

7.633.1 Detailed Description

This is a TR2 C++ Library header.

7.634 regex File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_REGEX`

7.634.1 Detailed Description

This is a TS C++ Library header.

7.635 regex File Reference

Macros

- `#define _GLIBCXX_REGEX`

7.635.1 Detailed Description

This is a Standard C++ Library header.

7.636 `scoped_allocator` File Reference

Classes

- class `std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs >`

Namespaces

- namespace `std`

Macros

- `#define _SCOPED_ALLOCATOR`

7.636.1 Detailed Description

This is a Standard C++ Library header.

7.637 `semaphore` File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_semaphore`
- `#define _GLIBCXX_SEMAPHORE`

Typedefs

- using `std::binary_semaphore` = `std::counting_semaphore< 1 >`

7.637.1 Detailed Description

This is a Standard C++ Library header.

7.638 `set` File Reference

Namespaces

- namespace `std`
- namespace `std::__debug`

Macros

- `#define _GLIBCXX_DEBUG_SET`

7.638.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.639 `set` File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define _GLIBCXX_EXPERIMENTAL_SET`

Typedefs

- `template<typename _Key, typename _Compare = less<_Key>>
using std::experimental::fundamentals_v2::pmr::multiset = std::multiset< _Key, _Compare, polymorphic_allocator< _Key > >`
- `template<typename _Key, typename _Compare = less<_Key>>
using std::experimental::fundamentals_v2::pmr::set = std::set< _Key, _Compare, polymorphic_allocator< _Key > >`

Functions

- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >
void std::experimental::erase_if (multiset< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >
void std::experimental::erase_if (set< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`

7.639.1 Detailed Description

This is a TS C++ Library header.

7.640 set File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_SET`

Typedefs

- `template<typename _Key, typename _Cmp = std::less<_Key>>
using std::pmr::multiset = std::multiset< _Key, _Cmp, polymorphic_allocator< _Key > >`
- `template<typename _Key, typename _Cmp = std::less<_Key>>
using std::pmr::set = std::set< _Key, _Cmp, polymorphic_allocator< _Key > >`

Functions

- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >
multiset< _Key, _Compare, _Alloc >::size_type std::erase_if (multiset< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >
set< _Key, _Compare, _Alloc >::size_type std::erase_if (set< _Key, _Compare, _Alloc > &__cont, _Predicate __pred)`

7.640.1 Detailed Description

This is a Standard C++ Library header.

7.641 `shared_mutex` File Reference

Classes

- class `std::shared_lock<_Mutex>`
- class `std::shared_mutex`
- class `std::shared_timed_mutex`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_shared_mutex`
- `#define __cpp_lib_shared_timed_mutex`
- `#define _GLIBCXX_SHARED_MUTEX`

7.641.1 Detailed Description

This is a Standard C++ Library header.

7.642 `source_location` File Reference

Classes

- struct `std::source_location`

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_source_location`
- `#define _GLIBCXX_SRCLOC`

7.642.1 Detailed Description

This is a Standard C++ Library header.

7.643 `span` File Reference

Namespaces

- namespace `std`
- namespace `std::__detail`

Macros

- `#define __cpp_lib_span`
- `#define _GLIBCXX_SPAN`

Functions

- `template<typename _Type, size_t _Extent>`
`span< const byte, _Extent==dynamic_extent ? dynamic_extent : _Extent *sizeof(_Type)> std::as_bytes (span< _Type, _Extent > __sp) noexcept`
- `template<typename _Type, size_t _Extent>`
`requires (!is_const_v<_Type>)`
`span< byte, _Extent==dynamic_extent ? dynamic_extent : _Extent *sizeof(_Type)> std::as_writable_bytes (span< _Type, _Extent > __sp) noexcept`
- `template<contiguous_iterator _Iter, typename _End >`
`std::span (_Iter, _End) -> span< remove_reference_t< iter_reference_t< _Iter > > >`
- `template<ranges::contiguous_range _Range>`
`std::span (_Range &&) -> span< remove_reference_t< ranges::range_reference_t< _Range & > > >`
- `template<typename _Type, size_t _ArrayExtent>`
`std::span (_Type(&)[_ArrayExtent]) -> span< _Type, _ArrayExtent >`
- `template<typename _Type, size_t _ArrayExtent>`
`std::span (array< _Type, _ArrayExtent > &) -> span< _Type, _ArrayExtent >`
- `template<typename _Type, size_t _ArrayExtent>`
`std::span (const array< _Type, _ArrayExtent > &) -> span< const _Type, _ArrayExtent >`

Variables

- `template<typename _Tp >`
`constexpr bool std::__detail::__is_span`
- `template<typename _Tp, size_t _Num>`
`constexpr bool std::__detail::__is_span< span< _Tp, _Num > >`
- `template<typename _Tp >`
`constexpr bool std::__detail::__is_std_array`
- `constexpr size_t std::dynamic_extent`
- `template<typename _ElementType, size_t _Extent>`
`constexpr bool std::ranges::enable_borrowed_range< span< _ElementType, _Extent > >`
- `template<typename _ElementType, size_t _Extent>`
`constexpr bool std::ranges::enable_view< span< _ElementType, _Extent > >`

7.643.1 Detailed Description

This is a Standard C++ Library header.

7.644 sstream File Reference

Classes

- class [std::basic_istringstream](#)< _CharT, _Traits, _Alloc >
- class [std::basic_ostringstream](#)< _CharT, _Traits, _Alloc >
- class [std::basic_stringbuf](#)< _CharT, _Traits, _Alloc >
- class [std::basic_stringstream](#)< _CharT, _Traits, _Alloc >

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_LVAL_REF_QUAL`
- `#define _GLIBCXX_SSTREAM`

Functions

- `template<class _CharT, class _Traits, class _Allocator >
void std::swap (basic_istream< _CharT, _Traits, _Allocator > &__x, basic_istream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >
void std::swap (basic_ostringstream< _CharT, _Traits, _Allocator > &__x, basic_ostringstream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >
void std::swap (basic_stringbuf< _CharT, _Traits, _Allocator > &__x, basic_stringbuf< _CharT, _Traits, _Allocator > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _CharT, class _Traits, class _Allocator >
void std::swap (basic_stringstream< _CharT, _Traits, _Allocator > &__x, basic_stringstream< _CharT, _Traits, _Allocator > &__y)`

7.644.1 Detailed Description

This is a Standard C++ Library header.

7.645 stack File Reference**Macros**

- `#define _GLIBCXX_STACK`

7.645.1 Detailed Description

This is a Standard C++ Library header.

7.646 stdatomic.h File Reference**7.646.1 Detailed Description**

This is a Standard C++ Library header.

7.647 stdexcept File Reference**Classes**

- class [std::domain_error](#)
- class [std::invalid_argument](#)
- class [std::length_error](#)
- class [std::logic_error](#)
- class [std::out_of_range](#)
- class [std::overflow_error](#)
- class [std::range_error](#)
- class [std::runtime_error](#)
- class [std::underflow_error](#)

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_STDEXCEPT`

Typedefs

- typedef [basic_string](#)< char > **std::__cow_string**
- typedef [basic_string](#)< char > **std::__sso_string**

7.647.1 Detailed Description

This is a Standard C++ Library header.

7.648 stdlib.h File Reference

7.648.1 Detailed Description

This is a Standard C++ Library header.

7.649 stop_token File Reference

Classes

- struct [std::nostopstate_t](#)
- class [std::stop_callback](#)< _Callback >
- class [std::stop_source](#)
- class [std::stop_token](#)

Namespaces

- namespace [std](#)

Macros

- #define **__cpp_lib_jthread**
- #define **_GLIBCXX_STOP_TOKEN**

Functions

- template<typename _Callback >
std::stop_callback ([stop_token](#), _Callback) -> [stop_callback](#)< _Callback >

Variables

- constexpr [nostopstate_t](#) **std::nostopstate**

7.649.1 Detailed Description

This is a Standard C++ Library header.

7.650 streambuf File Reference

Classes

- class [std::basic_streambuf](#)< _CharT, _Traits >

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBXX_STREAMBUF`
- `#define _IsUnused`

Functions

- `template<typename _CharT, typename _Traits >`
`streamsize std::__copy_streambufs_eof (basic_streambuf< _CharT, _Traits > *, basic_streambuf< _CharT, _Traits > *, bool &)`
- `template<> streamsize std::__copy_streambufs_eof (basic_streambuf< char > * __sbin, basic_streambuf< char > * __sbout, bool & __ineof)`
- `template<> streamsize std::__copy_streambufs_eof (basic_streambuf< wchar_t > * __sbin, basic_streambuf< wchar_t > * __sbout, bool & __ineof)`

7.650.1 Detailed Description

This is a Standard C++ Library header.

7.651 string File Reference

Classes

- class `__gnu_debug::basic_string< _CharT, _Traits, _Allocator >`
- struct `std::hash< __gnu_debug::basic_string< _CharT > >`

Namespaces

- namespace `__gnu_debug`
- namespace `std`

Macros

- `#define __glibcxx_check_string_constructor(_Str)`
- `#define __glibcxx_check_string_n_constructor(_Str, _Size)`
- `#define _GLIBCXX_DEBUG_STRING`
- `#define _GLIBCXX_DEBUG_VERIFY_STR_COND_AT(_Cond, _File, _Line, _Func)`
- `#define _GLIBCXX_INSERT_RETURNS_ITERATOR`
- `#define _GLIBCXX_INSERT_RETURNS_ITERATOR_ONLY(expr)`

Typedefs

- `typedef basic_string< char > __gnu_debug::string`
- `typedef basic_string< char16_t > __gnu_debug::u16string`
- `typedef basic_string< char32_t > __gnu_debug::u32string`
- `typedef basic_string< wchar_t > __gnu_debug::wstring`

Functions

- `template<typename _CharT, typename _Integer >`
`const _CharT * __gnu_debug::__check_string (const _CharT * __s, _Integer __n, const char * __file, unsigned int __line, const char * __function)`
- `template<typename _CharT >`
`const _CharT * __gnu_debug::__check_string (const _CharT * __s, const char * __file, unsigned int __line, const char * __function)`

- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator== (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__↵`
`__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator== (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string<`
`_CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator> (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__↵`
`rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator> (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__↵`
`rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator> (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string<`
`_CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator>= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__↵`
`__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator>= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__↵`
`__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool __gnu_debug::operator>= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string<`
`_CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > & __gnu_debug::operator>> (std::basic_istream< _CharT, _Traits >`
`&__is, basic_string< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`void __gnu_debug::swap (basic_string< _CharT, _Traits, _Allocator > &__lhs, basic_string< _CharT, _Traits,`
`_Allocator > &__rhs)`

7.651.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.652 string File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_STRING`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Up >`
`void std::experimental::erase (basic_string< _CharT, _Traits, _Alloc > &__cont, const _Up &__value)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Predicate >`
`void std::experimental::erase_if (basic_string< _CharT, _Traits, _Alloc > &__cont, _Predicate __pred)`

7.652.1 Detailed Description

This is a TS C++ Library header.

7.653 string File Reference

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_erase_if`
- `#define _GLIBCXX_STRING`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Up >`
`constexpr basic_string< _CharT, _Traits, _Alloc >::size_type std::erase (basic_string< _CharT, _Traits, _Alloc > &__cont, const _Up &__value)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Predicate >`
`constexpr basic_string< _CharT, _Traits, _Alloc >::size_type std::erase_if (basic_string< _CharT, _Traits, _Alloc > &__cont, _Predicate __pred)`

7.653.1 Detailed Description

This is a Standard C++ Library header.

7.654 string_view File Reference

Classes

- class [std::experimental::fundamentals_v1::basic_string_view](#)< _CharT, _Traits >

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_string_view`
- `#define _GLIBCXX_EXPERIMENTAL_STRING_VIEW`

Typedefs

- using `std::experimental::string_view = basic_string_view< char >`
- using `std::experimental::u16string_view = basic_string_view< char16_t >`
- using `std::experimental::u32string_view = basic_string_view< char32_t >`
- using `std::experimental::wstring_view = basic_string_view< wchar_t >`

Functions

- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator!= (__type_identity_t< basic_string_view< _CharT, _Traits > > <←`
`__x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator!= (basic_string_view< _CharT, _Traits > __x, __type_identity_t<`
`basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator!= (basic_string_view< _CharT, _Traits > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `constexpr basic_string_view< char > std::experimental::literals::operator""sv (const char *__str, size_t __len)`
`noexcept`
- `constexpr basic_string_view< char16_t > std::experimental::literals::operator""sv (const char16_t *__str,`
`size_t __len) noexcept`
- `constexpr basic_string_view< char32_t > std::experimental::literals::operator""sv (const char32_t *__str,`
`size_t __len) noexcept`
- `constexpr basic_string_view< wchar_t > std::experimental::literals::operator""sv (const wchar_t *__str,`
`size_t __len) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator< (__type_identity_t< basic_string_view< _CharT, _Traits > > <←`
`__x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator< (basic_string_view< _CharT, _Traits > __x, __type_identity_t<`
`basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator< (basic_string_view< _CharT, _Traits > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::experimental::operator<< (basic_ostream< _CharT, _Traits > & <←`
`__os, basic_string_view< _CharT, _Traits > __str)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator<= (__type_identity_t< basic_string_view< _CharT, _Traits > >`
`__x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator<= (basic_string_view< _CharT, _Traits > __x, __type_identity_t<`
`basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator<= (basic_string_view< _CharT, _Traits > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator== (__type_identity_t< basic_string_view< _CharT, _Traits > > <←`
`__x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator== (basic_string_view< _CharT, _Traits > __x, __type_identity_t<`
`basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator== (basic_string_view< _CharT, _Traits > __x, basic_string_view<`
`_CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator> (__type_identity_t< basic_string_view< _CharT, _Traits > > <←`
`__x, basic_string_view< _CharT, _Traits > __y) noexcept`

- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator> (basic_string_view< _CharT, _Traits > __x, __type_identity_t<
basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator> (basic_string_view< _CharT, _Traits > __x, basic_string_view<
 _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator>= (__type_identity_t< basic_string_view< _CharT, _Traits > >
 __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator>= (basic_string_view< _CharT, _Traits > __x, __type_identity_t<
basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::experimental::operator>= (basic_string_view< _CharT, _Traits > __x, basic_string_view<
 _CharT, _Traits > __y) noexcept`

7.654.1 Detailed Description

This is a TS C++ Library header.

7.655 `string_view` File Reference

Classes

- class [std::basic_string_view](#)< _CharT, _Traits >

Namespaces

- namespace [std](#)
- namespace [std::literals](#)

Macros

- `#define __cpp_lib_constexpr_string_view`
- `#define __cpp_lib_starts_ends_with`
- `#define __cpp_lib_string_view`
- `#define _GLIBCXX_STRING_VIEW`

Typedefs

- using `std::string_view` = [basic_string_view](#)< char >
- using `std::u16string_view` = [basic_string_view](#)< char16_t >
- using `std::u32string_view` = [basic_string_view](#)< char32_t >
- using `std::wstring_view` = [basic_string_view](#)< wchar_t >

Functions

- `constexpr size_t std::__sv_check (size_t __size, size_t __pos, const char *__s)`
- `constexpr size_t std::__sv_limit (size_t __size, size_t __pos, size_t __off) noexcept`
- `template<contiguous_iterator _It, sized_sentinel_for< _It > _End>`
`std::basic_string_view (_It, _End) -> basic_string_view< iter_value_t< _It > >`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator!= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`

- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator!= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator!= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `constexpr basic_string_view< char > std::literals::operator""sv (const char * __str, size_t __len) noexcept`
- `constexpr basic_string_view< char16_t > std::literals::operator""sv (const char16_t * __str, size_t __len) noexcept`
- `constexpr basic_string_view< char32_t > std::literals::operator""sv (const char32_t * __str, size_t __len) noexcept`
- `constexpr basic_string_view< wchar_t > std::literals::operator""sv (const wchar_t * __str, size_t __len) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator< (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator< (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator< (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > & __os, basic_string_view< _CharT, _Traits > __str)`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator<= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator<= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator<= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator== (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator== (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator== (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator> (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator> (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator> (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`

- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator>= (__type_identity_t< basic_string_view< _CharT, _Traits > > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator>= (basic_string_view< _CharT, _Traits > __x, __type_identity_t< basic_string_view< _CharT, _Traits > > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::operator>= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`

Variables

- `template<typename _CharT, typename _Traits >`
`constexpr bool std::ranges::enable_borrowed_range< basic_string_view< _CharT, _Traits > >`
- `template<typename _CharT, typename _Traits >`
`constexpr bool std::ranges::enable_view< basic_string_view< _CharT, _Traits > >`

7.655.1 Detailed Description

This is a Standard C++ Library header.

7.656 [syncstream](#) File Reference

Macros

- `#define _GLIBCXX_SYNCSTREAM`

7.656.1 Detailed Description

This is a Standard C++ Library header.

7.657 [system_error](#) File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_SYSTEM_ERROR`

Variables

- `template<typename _Tp >`
`constexpr bool std::experimental::is_error_code_enum_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_error_condition_enum_v`

7.657.1 Detailed Description

This is a TS C++ Library header.

7.658 `system_error` File Reference

Classes

- class `std::_V2::error_category`
- class `std::error_code`
- class `std::error_condition`
- struct `std::hash< error_code >`
- struct `std::hash< error_condition >`
- struct `std::is_error_code_enum< _Tp >`
- struct `std::is_error_condition_enum< _Tp >`
- class `std::system_error`

Namespaces

- namespace `std`

Macros

- `#define _GLIBCXX_SYSTEM_ERROR`

Functions

- const `error_category` & `std::generic_category` () noexcept
- void `std::__adl_only::make_error_code` ()=delete
- void `std::__adl_only::make_error_condition` ()=delete
- const `error_category` & `std::system_category` () noexcept

Variables

- template<typename _Tp >
constexpr bool `std::is_error_code_enum_v`
- template<typename _Tp >
constexpr bool `std::is_error_condition_enum_v`

7.658.1 Detailed Description

This is a Standard C++ Library header.

7.659 `tgmath.h` File Reference

Macros

- `#define _GLIBCXX_TGMATH_H`

7.659.1 Detailed Description

This is a Standard C++ Library header.

7.660 `thread` File Reference

Classes

- class `std::jthread`

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_THREAD`

7.660.1 Detailed Description

This is a Standard C++ Library header.

7.661 `bool_set` File Reference

Classes

- class [std::tr2::bool_set](#)

Namespaces

- namespace [std](#)
- namespace [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_BOOL_SET`

Functions

- `bool std::tr2::certainly (bool_set __b)`
- `bool std::tr2::contains (bool_set __s, bool_set __t)`
- `bool std::tr2::equals (bool_set __s, bool_set __t)`
- `bool std::tr2::is_emptyset (bool_set __b)`
- `bool std::tr2::is_indeterminate (bool_set __b)`
- `bool std::tr2::is_singleton (bool_set __b)`
- `bool_set std::tr2::operator!= (bool __s, bool_set __t)`
- `bool_set std::tr2::operator!= (bool_set __s, bool __t)`
- `bool_set std::tr2::operator!= (bool_set __s, bool_set __t)`
- `bool_set std::tr2::operator& (bool __s, bool_set __t)`
- `bool_set std::tr2::operator& (bool_set __s, bool __t)`
- `bool_set std::tr2::operator== (bool __s, bool_set __t)`
- `bool_set std::tr2::operator== (bool_set __s, bool __t)`
- `bool_set std::tr2::operator^ (bool __s, bool_set __t)`
- `bool_set std::tr2::operator^ (bool_set __s, bool __t)`
- `bool_set std::tr2::operator| (bool __s, bool_set __t)`
- `bool_set std::tr2::operator| (bool_set __s, bool __t)`
- `bool std::tr2::possibly (bool_set __b)`
- `bool_set std::tr2::set_complement (bool_set __b)`
- `bool_set std::tr2::set_intersection (bool __s, bool_set __t)`
- `bool_set std::tr2::set_intersection (bool_set __s, bool __t)`
- `bool_set std::tr2::set_intersection (bool_set __s, bool_set __t)`
- `bool_set std::tr2::set_union (bool __s, bool_set __t)`
- `bool_set std::tr2::set_union (bool_set __s, bool __t)`
- `bool_set std::tr2::set_union (bool_set __s, bool_set __t)`

7.661.1 Detailed Description

This is a TR2 C++ Library header.

7.662 bool_set.tcc File Reference

Namespaces

- namespace [std](#)
- namespace [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_BOOL_SET_TCC`

7.662.1 Detailed Description

This is a TR2 C++ Library header.

7.663 dynamic_bitset File Reference

Classes

- struct [std::tr2::__dynamic_bitset_base<_WordT, _Alloc>](#)
- class [std::tr2::dynamic_bitset<_WordT, _Alloc>](#)
- class [std::tr2::dynamic_bitset<_WordT, _Alloc>::reference](#)

Namespaces

- namespace [std](#)
- namespace [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_DYNAMIC_BITSET`

Functions

- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc>
std::basic_ostream<_CharT, _Traits> & std::tr2::operator<< (std::basic_ostream<_CharT, _Traits> &__os,
const dynamic_bitset<_WordT, _Alloc> &__x)`
- `template<typename _WordT, typename _Alloc>
bool std::tr2::operator!= (const dynamic_bitset<_WordT, _Alloc> &__lhs, const dynamic_bitset<_WordT, _Alloc>
& __rhs)`
- `template<typename _WordT, typename _Alloc>
bool std::tr2::operator<= (const dynamic_bitset<_WordT, _Alloc> &__lhs, const dynamic_bitset<_WordT, _Alloc> &__rhs)`
- `template<typename _WordT, typename _Alloc>
bool std::tr2::operator> (const dynamic_bitset<_WordT, _Alloc> &__lhs, const dynamic_bitset<_WordT, _Alloc>
& __rhs)`
- `template<typename _WordT, typename _Alloc>
bool std::tr2::operator>= (const dynamic_bitset<_WordT, _Alloc> &__lhs, const dynamic_bitset<_WordT, _Alloc> &__rhs)`

- `template<typename _WordT , typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator& (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT , typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator- (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT , typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator^ (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT , typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator| (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`

7.663.1 Detailed Description

This is a TR2 C++ Library header.

7.664 dynamic_bitset.tcc File Reference

Namespaces

- namespace `std`
- namespace `std::tr2`

Macros

- `#define _GLIBCXX_TR2_DYNAMIC_BITSET_TCC`

Functions

- `template<typename _CharT , typename _Traits , typename _WordT , typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & std::tr2::operator>> (std::basic_istream< _CharT, _Traits > &__is, dynamic_bitset< _WordT, _Alloc > &__x)`

7.664.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<tr2/dynamic_bitset>`.

7.665 tuple File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define __cpp_lib_experimental_tuple`
- `#define _GLIBCXX_EXPERIMENTAL_TUPLE`

Functions

- `template<typename _Fn, typename _Tuple, std::size_t... _Idx>`
`constexpr decltype(auto) std::experimental::__apply_impl (_Fn &&__f, _Tuple &&__t, std::index_sequence<_Idx... >)`
- `template<typename _Fn, typename _Tuple >`
`constexpr decltype(auto) std::experimental::apply (_Fn &&__f, _Tuple &&__t)`

Variables

- `template<typename _Tp >`
`constexpr size_t std::experimental::tuple_size_v`

7.665.1 Detailed Description

This is a TS C++ Library header.

7.666 tuple File Reference

Classes

- struct [std::_Tuple_impl](#)< _Idx, _Head, _Tail... >
- class [std::tuple](#)< _Elements >
- class [std::tuple](#)< _T1, _T2 >
- struct [std::tuple_element](#)< __i, [tuple](#)< _Types... > >
- struct [std::tuple_size](#)< [tuple](#)< _Elements... > >
- struct [std::uses_allocator](#)< [tuple](#)< _Types... >, _Alloc >

Namespaces

- namespace [std](#)

Macros

- `#define __cpp_lib_apply`
- `#define __cpp_lib_constexpr_tuple`
- `#define __cpp_lib_make_from_tuple`
- `#define __cpp_lib_tuples_by_type`
- `#define _GLIBCXX_TUPLE`

Typedefs

- `template<typename _Tp >`
`using std::__empty_not_final = __conditional_t< __is_final(_Tp), false_type, __is_empty_non_tuple< _Tp >`
`>`

Functions

- `template<typename _Fn, typename _Tuple, size_t... _Idx>`
`constexpr decltype(auto) std::__apply_impl (_Fn &&__f, _Tuple &&__t, index_sequence< _Idx... >)`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & std::__get_helper (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & std::__get_helper (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`

- `template<size_t __i, typename... _Types>
__enable_if_t<(__i >= sizeof...(_Types))> std::__get_helper (const tuple< _Types... > &)=delete`
- `template<typename _Tp, typename _Tuple, size_t... _Idx>
constexpr _Tp std::__make_from_tuple_impl (_Tuple &&__t, index_sequence< _Idx... >)`
- `template<typename _Fn, typename _Tuple >
constexpr decltype(auto) std::apply (_Fn &&__f, _Tuple &&__t) noexcept(__unpack_std_tuple< is_nothrow_invocable, _Fn, _Tuple >)`
- `template<typename... _Elements>
constexpr tuple< _Elements &&... > std::forward_as_tuple (_Elements &&... __args) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp && std::get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp & std::get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr const _Tp && std::get (const array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr const _Tp & std::get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<size_t __i, typename... _Elements>
constexpr const __tuple_element_t< __i, tuple< _Elements... > > && std::get (const tuple< _Elements... > &&__t) noexcept`
- `template<size_t __i, typename... _Elements>
constexpr const __tuple_element_t< __i, tuple< _Elements... > > & std::get (const tuple< _Elements... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>
constexpr const _Tp && std::get (const tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>
constexpr const _Tp & std::get (const tuple< _Types... > &__t) noexcept`
- `template<size_t __i, typename... _Elements>
constexpr __tuple_element_t< __i, tuple< _Elements... > > && std::get (tuple< _Elements... > &&__t) noexcept`
- `template<size_t __i, typename... _Elements>
constexpr __tuple_element_t< __i, tuple< _Elements... > > & std::get (tuple< _Elements... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>
constexpr _Tp && std::get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>
constexpr _Tp & std::get (tuple< _Types... > &__t) noexcept`
- `template<typename _Tp, typename _Tuple >
constexpr _Tp std::make_from_tuple (_Tuple &&__t) noexcept(__unpack_std_tuple< is_nothrow_constructible, _Tp, _Tuple >)`
- `template<typename... _Elements>
constexpr tuple< typename __decay_and_strip< _Elements >::__type... > std::make_tuple (_Elements &&... __args)`
- `template<typename... _TElements, typename... _UElements>
constexpr bool std::operator!= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>
constexpr bool std::operator< (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>
constexpr bool std::operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>
constexpr bool std::operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>
constexpr bool std::operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`

- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _Elements>`
`constexpr enable_if<!__and< __is_swappable< _Elements >... >::value >::type std::swap (tuple< _Elements... > &, tuple< _Elements... > &)=delete`
- `template<typename... _Elements>`
`constexpr enable_if< __and< __is_swappable< _Elements >... >::value >::type std::swap (tuple< _Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &... > std::tie (_Elements &... __args) noexcept`
- `template<typename... _UTypes>`
`std::tuple (_UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename... _UTypes>`
`std::tuple (allocator_arg_t, _Alloc, _UTypes...) -> tuple< _UTypes... >`
- `template<typename _Alloc, typename _T1, typename _T2 >`
`std::tuple (allocator_arg_t, _Alloc, pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename _Alloc, typename... _UTypes>`
`std::tuple (allocator_arg_t, _Alloc, tuple< _UTypes... >) -> tuple< _UTypes... >`
- `template<typename _T1, typename _T2 >`
`std::tuple (pair< _T1, _T2 >) -> tuple< _T1, _T2 >`
- `template<typename... _Tpls, typename = typename enable_if< __and< __is_tuple_like< _Tpls>...>::value>::type>`
`constexpr auto std::tuple_cat (_Tpls &&... __tpls) -> typename __tuple_cat_result< _Tpls... >::type`

Variables

- `template<template< typename... > class _Trait, typename _Tp, typename _Tuple >`
`constexpr bool std::__unpack_std_tuple`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > & >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, const tuple< _Up... > >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, tuple< _Up... > & >`
- `template<template< typename... > class _Trait, typename _Tp, typename... _Up>`
`constexpr bool std::__unpack_std_tuple< _Trait, _Tp, tuple< _Up... > >`
- `constexpr _Swallow_assign std::ignore`
- `template<typename... _Types>`
`constexpr size_t std::tuple_size_v< const tuple< _Types... > >`
- `template<typename... _Types>`
`constexpr size_t std::tuple_size_v< tuple< _Types... > >`

7.666.1 Detailed Description

This is a Standard C++ Library header.

7.667 type_traits File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define _GLIBCXX_EXPERIMENTAL_TYPE_TRAITS`
- `#define __cpp_lib_experimental_type_trait_variable_templates`
- `template<typename _Tp >`
`constexpr size_t std::experimental::alignment_of_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::experimental::extent_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_abstract_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_array_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_assignable_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::experimental::is_base_of_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_const_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_constructible_v`
- `template<typename _From, typename _To >`
`constexpr bool std::experimental::is_convertible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_empty_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_integral_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_nothrow_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_nothrow_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_null_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_rvalue_reference_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_same_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_standard_layout_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivial_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::is_trivially_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::is_trivially_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_trivially_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_unsigned_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_void_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::is_volatile_v`
- `template<typename _Tp >`
`constexpr size_t std::experimental::rank_v`
- `#define __cpp_lib_experimental_detect`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
`using std::experimental::detected_or = std::__detected_or< _Default, _Op, _Args... >`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
`using std::experimental::detected_or_t = typename detected_or< _Default, _Op, _Args... >::type`
- `template<template< typename... > class _Op, typename... _Args>`
`using std::experimental::detected_t = typename std::__detector< nonesuch, void, _Op, _Args... >::type`
- `template<template< typename... > class _Op, typename... _Args>`
`using std::experimental::is_detected = typename std::__detector< nonesuch, void, _Op, _Args... >::value_t`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
`using std::experimental::is_detected_convertible = is_convertible< detected_t< _Op, _Args... >, _To >`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::is_detected_convertible_v`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
`using std::experimental::is_detected_exact = is_same< _Expected, detected_t< _Op, _Args... > >`
- `template<typename _Expected, template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::is_detected_exact_v`
- `template<template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::is_detected_v`
- `template<typename... >`
`using std::experimental::void_t = void`

- `#define __cpp_lib_experimental_logical_traits`
- `template<typename... _Bn>`
`constexpr bool std::experimental::conjunction_v`
- `template<typename... _Bn>`
`constexpr bool std::experimental::disjunction_v`
- `template<typename _Pp >`
`constexpr bool std::experimental::negation_v`

7.667.1 Detailed Description

This is a TS C++ Library header.

This header defines variable templates for the C++14 type traits.

Equivalent variable templates are defined in namespace `std` since C++17.

See also

[variable_templates](#)

Since

C++14

7.668 `type_traits` File Reference

Classes

- struct [std::tr2::__reflection_typelist<_First, _Rest... >](#)
- struct [std::tr2::__reflection_typelist<>](#)
- struct [std::tr2::bases<_Tp >](#)
- struct [std::tr2::direct_bases<_Tp >](#)

Namespaces

- namespace [std](#)
- namespace [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_TYPE_TRAITS`

7.668.1 Detailed Description

This is a TR2 C++ Library header.

7.669 `type_traits` File Reference

Classes

- struct [std::__is_nullptr_t<_Tp >](#)
- struct [std::add_const<_Tp >](#)
- struct [std::add_cv<_Tp >](#)
- struct [std::add_lvalue_reference<_Tp >](#)
- struct [std::add_pointer<_Tp >](#)
- struct [std::add_rvalue_reference<_Tp >](#)
- struct [std::add_volatile<_Tp >](#)
- struct [std::aligned_storage<_Len, _Align >](#)

- struct `std::aligned_union< _Len, _Types >`
- struct `std::alignment_of< _Tp >`
- struct `std::conditional< _Cond, _Iftrue, _Iffalse >`
- class `std::decay< _Tp >`
- struct `std::enable_if< bool, _Tp >`
- struct `std::extent< typename, _UInt >`
- struct `std::has_unique_object_representations< _Tp >`
- struct `std::has_virtual_destructor< _Tp >`
- struct `std::integral_constant< _Tp, __v >`
- struct `std::invoke_result< _Functor, _ArgTypes >`
- struct `std::is_abstract< _Tp >`
- struct `std::is_aggregate< _Tp >`
- struct `std::is_arithmetic< _Tp >`
- struct `std::is_array< typename >`
- struct `std::is_assignable< _Tp, _Up >`
- struct `std::is_base_of< _Base, _Derived >`
- struct `std::is_bounded_array< _Tp >`
- struct `std::is_class< _Tp >`
- struct `std::is_compound< _Tp >`
- struct `std::is_const< typename >`
- struct `std::is_constructible< _Tp, _Args >`
- struct `std::is_convertible< _From, _To >`
- struct `std::is_copy_assignable< _Tp >`
- struct `std::is_copy_constructible< _Tp >`
- struct `std::is_default_constructible< _Tp >`
- struct `std::is_destructible< _Tp >`
- struct `std::is_empty< _Tp >`
- struct `std::is_enum< _Tp >`
- struct `std::is_final< _Tp >`
- struct `std::is_floating_point< _Tp >`
- struct `std::is_function< _Tp >`
- struct `std::is_fundamental< _Tp >`
- struct `std::is_integral< _Tp >`
- struct `std::is_invocable< _Fn, _ArgTypes >`
- struct `std::is_invocable_r< _Ret, _Fn, _ArgTypes >`
- struct `std::is_layout_compatible< _Tp, _Up >`
- struct `std::is_literal_type< _Tp >`
- struct `std::is_lvalue_reference< typename >`
- struct `std::is_member_function_pointer< _Tp >`
- struct `std::is_member_object_pointer< _Tp >`
- struct `std::is_member_pointer< _Tp >`
- struct `std::is_move_assignable< _Tp >`
- struct `std::is_move_constructible< _Tp >`
- struct `std::is_nothrow_assignable< _Tp, _Up >`
- struct `std::is_nothrow_constructible< _Tp, _Args >`
- struct `std::is_nothrow_convertible< _From, _To >`
- struct `std::is_nothrow_copy_assignable< _Tp >`
- struct `std::is_nothrow_copy_constructible< _Tp >`
- struct `std::is_nothrow_default_constructible< _Tp >`
- struct `std::is_nothrow_destructible< _Tp >`
- struct `std::is_nothrow_invocable< _Fn, _ArgTypes >`

- struct `std::is_nothrow_invocable_r< _Ret, _Fn, _ArgTypes >`
- struct `std::is_nothrow_move_assignable< _Tp >`
- struct `std::is_nothrow_move_constructible< _Tp >`
- struct `std::is_nothrow_swappable< _Tp >`
- struct `std::is_nothrow_swappable_with< _Tp, _Up >`
- struct `std::is_null_pointer< _Tp >`
- struct `std::is_object< _Tp >`
- struct `std::is_pod< _Tp >`
- struct `std::is_pointer< _Tp >`
- struct `std::is_pointer_interconvertible_base_of< _Base, _Derived >`
- struct `std::is_polymorphic< _Tp >`
- struct `std::is_reference< _Tp >`
- struct `std::is_rvalue_reference< typename >`
- struct `std::is_same< _Tp, _Up >`
- struct `std::is_scalar< _Tp >`
- struct `std::is_signed< _Tp >`
- struct `std::is_standard_layout< _Tp >`
- struct `std::is_swappable< _Tp >`
- struct `std::is_swappable_with< _Tp, _Up >`
- struct `std::is_trivial< _Tp >`
- struct `std::is_trivially_assignable< _Tp, _Up >`
- struct `std::is_trivially_constructible< _Tp, _Args >`
- struct `std::is_trivially_copy_assignable< _Tp >`
- struct `std::is_trivially_copy_constructible< _Tp >`
- struct `std::is_trivially_copyable< _Tp >`
- struct `std::is_trivially_default_constructible< _Tp >`
- struct `std::is_trivially_destructible< _Tp >`
- struct `std::is_trivially_move_assignable< _Tp >`
- struct `std::is_trivially_move_constructible< _Tp >`
- struct `std::is_unbounded_array< _Tp >`
- struct `std::is_union< _Tp >`
- struct `std::is_unsigned< _Tp >`
- struct `std::is_void< _Tp >`
- struct `std::is_volatile< typename >`
- struct `std::make_signed< _Tp >`
- struct `std::make_unsigned< _Tp >`
- struct `std::rank< typename >`
- struct `std::remove_all_extents< _Tp >`
- struct `std::remove_const< _Tp >`
- struct `std::remove_cv< _Tp >`
- struct `std::remove_extent< _Tp >`
- struct `std::remove_pointer< _Tp >`
- struct `std::remove_reference< _Tp >`
- struct `std::remove_volatile< _Tp >`
- struct `std::underlying_type< _Tp >`
- struct `std::unwrap_ref_decay< _Tp >`
- struct `std::unwrap_reference< _Tp >`

Namespaces

- namespace `std`

Macros

- #define `__cpp_lib_bool_constant`
- #define `__cpp_lib_bounded_array_traits`
- #define `__cpp_lib_has_unique_object_representations`
- #define `__cpp_lib_integral_constant_callable`
- #define `__cpp_lib_is_aggregate`
- #define `__cpp_lib_is_constant_evaluated`
- #define `__cpp_lib_is_final`
- #define `__cpp_lib_is_invocable`
- #define `__cpp_lib_is_layout_compatible`
- #define `__cpp_lib_is_nothrow_convertible`
- #define `__cpp_lib_is_null_pointer`
- #define `__cpp_lib_is_pointer_interconvertible`
- #define `__cpp_lib_is_swappable`
- #define `__cpp_lib_logical_traits`
- #define `__cpp_lib_result_of_sfinae`
- #define `__cpp_lib_transformation_trait_aliases`
- #define `__cpp_lib_type_trait_variable_templates`
- #define `__cpp_lib_unwrap_ref`
- #define `__cpp_lib_void_t`
- #define `_GLIBCXX_TYPE_TRAITS`

Typedefs

- template<bool `_Cond`, typename `_If`, typename `_Else` >
using `std::conditional_t` = typename `__conditional<_Cond>::template type<_If, _Else>`
- template<typename `_ToElementType`, typename `_FromElementType` >
using `std::is_array_convertible` = `is_convertible<_FromElementType(*)[], _ToElementType(*)[]>`
- template<typename `_Tp`, typename `_Up` >
using `std::is_nothrow_assignable_impl` = `__bool_constant<__is_nothrow_assignable(_Tp, _Up)>`
- template<typename `_Tp` >
using `std::add_const_t` = typename `add_const<_Tp>::type`
- template<typename `_Tp` >
using `std::add_cv_t` = typename `add_cv<_Tp>::type`
- template<typename `_Tp` >
using `std::add_lvalue_reference_t` = typename `add_lvalue_reference<_Tp>::type`
- template<typename `_Tp` >
using `std::add_pointer_t` = typename `add_pointer<_Tp>::type`
- template<typename `_Tp` >
using `std::add_rvalue_reference_t` = typename `add_rvalue_reference<_Tp>::type`
- template<typename `_Tp` >
using `std::add_volatile_t` = typename `add_volatile<_Tp>::type`
- template<size_t `_Len`, size_t `_Align` = `__alignof__(typename __aligned_storage_msa<_Len>::type)>` >
using `std::aligned_storage_t` = typename `aligned_storage<_Len, _Align>::type`
- template<size_t `_Len`, typename... `_Types` >
using `std::aligned_union_t` = typename `aligned_union<_Len, _Types...>::type`
- template<bool `__v` >
using `std::bool_constant` = `integral_constant<bool, __v>`
- template<typename... `_Tp` >
using `std::common_reference_t` = typename `common_reference<_Tp...>::type`
- template<typename... `_Tp` >
using `std::common_type_t` = typename `common_type<_Tp...>::type`

- `template<bool _Cond, typename _Iftrue, typename _Iffalse >`
`using std::conditional_t = typename conditional< _Cond, _Iftrue, _Iffalse >::type`
- `template<typename _Tp >`
`using std::decay_t = typename decay< _Tp >::type`
- `template<bool _Cond, typename _Tp = void>`
`using std::enable_if_t = typename enable_if< _Cond, _Tp >::type`
- `using std::false_type = integral_constant< bool, false >`
- `template<typename _Fn, typename... _Args>`
`using std::invoke_result_t = typename invoke_result< _Fn, _Args... >::type`
- `template<typename _Tp >`
`using std::make_signed_t = typename make_signed< _Tp >::type`
- `template<typename _Tp >`
`using std::make_unsigned_t = typename make_unsigned< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_all_extents_t = typename remove_all_extents< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_const_t = typename remove_const< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_cv_t = typename remove_cv< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_extent_t = typename remove_extent< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_pointer_t = typename remove_pointer< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_reference_t = typename remove_reference< _Tp >::type`
- `template<typename _Tp >`
`using std::remove_volatile_t = typename remove_volatile< _Tp >::type`
- `template<typename _Tp >`
`using std::result_of_t = typename result_of< _Tp >::type`
- `using std::true_type = integral_constant< bool, true >`
- `template<typename _Tp >`
`using std::underlying_type_t = typename underlying_type< _Tp >::type`
- `template<typename... >`
`using std::void_t = void`

- `template<typename _Tp >`
`using std::unwrap_reference_t = typename unwrap_reference< _Tp >::type`

- `template<typename _Tp >`
`using std::unwrap_ref_decay_t = typename unwrap_ref_decay< _Tp >::type`

Functions

- `template<typename _Tp >`
`auto std::declval () noexcept -> decltype(__declval< _Tp > (0))`
- `constexpr bool std::is_constant_evaluated () noexcept`
- `template<typename _S1, typename _S2, typename _M1, typename _M2 >`
`constexpr bool std::is_corresponding_member (_M1 _S1::* __m1, _M2 _S2::* __m2) noexcept`
- `template<typename _Tp, typename _Mem >`
`constexpr bool std::is_pointer_interconvertible_with_class (_Mem _Tp::* __mp) noexcept`

- `template<typename _Tp >`
`constexpr __Require<__not_<__is_tuple_like<_Tp>>, is_move_constructible<_Tp>, is_move_assignable<`
`_Tp>> std::swap(_Tp&, _Tp&) noexcept(__and_<is_nothrow_move_constructible<_Tp>, is_nothrow_move_assignable<`
`_Tp>>::value)`
- `template<typename _Tp, size_t _Nm>`
`constexpr __enable_if_t<__is_swappable<_Tp>::value> std::swap(_Tp(&__a)[_Nm], _Tp(&__b)[_Nm])`
`noexcept(__is_nothrow_swappable<_Tp>::value)`

Variables

- `template<typename _Tp >`
`constexpr bool std::has_unique_object_representations_v`
- `template<typename _Tp >`
`constexpr bool std::is_aggregate_v`
- `template<typename _Tp >`
`constexpr bool std::is_bounded_array_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_layout_compatible_v`
- `template<typename _From, typename _To >`
`constexpr bool std::is_nothrow_convertible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_swappable_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_nothrow_swappable_with_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::is_pointer_interconvertible_base_of_v`
- `template<typename _Tp >`
`constexpr bool std::is_swappable_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_swappable_with_v`
- `template<typename _Tp >`
`constexpr bool std::is_unbounded_array_v`
- `template<typename... _Bn>`
`constexpr bool std::conjunction_v`
- `template<typename... _Bn>`
`constexpr bool std::disjunction_v`
- `template<typename _Pp >`
`constexpr bool std::negation_v`
- `template<typename _Tp >`
`constexpr size_t std::alignment_of_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::extent_v`
- `template<typename _Tp >`
`constexpr bool std::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr bool std::is_abstract_v`
- `template<typename _Tp >`
`constexpr bool std::is_arithmetic_v`

- `template<typename _Tp >`
`constexpr bool std::is_array_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_assignable_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::is_base_of_v`
- `template<typename _Tp >`
`constexpr bool std::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::is_const_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_constructible_v`
- `template<typename _From, typename _To >`
`constexpr bool std::is_convertible_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_empty_v`
- `template<typename _Tp >`
`constexpr bool std::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool std::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::is_integral_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_r_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_invocable_v`
- `template<typename _Tp >`
`constexpr bool std::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_function_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_member_pointer_v`

- `template<typename _Tp >`
`constexpr bool std::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_nothrow_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_destructible_v`
- `template<typename _Ret, typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_r_v`
- `template<typename _Fn, typename... _Args>`
`constexpr bool std::is_nothrow_invocable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_nothrow_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_null_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool std::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::is_rvalue_reference_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_same_v`
- `template<typename _Tp >`
`constexpr bool std::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivial_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::is_trivially_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::is_trivially_constructible_v`

- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::is_trivially_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::is_unsigned_v`
- `template<typename _Tp >`
`constexpr bool std::is_void_v`
- `template<typename _Tp >`
`constexpr bool std::is_volatile_v`
- `template<typename _Tp >`
`constexpr size_t std::rank_v`
- `#define __cpp_lib_remove_cvref`
- `template<typename _Tp >`
`using std::remove_cvref_t = typename remove_cvref< _Tp >::type`
- `#define __cpp_lib_type_identity`
- `template<typename _Tp >`
`using std::type_identity_t = typename type_identity< _Tp >::type`

7.669.1 Detailed Description

This is a Standard C++ Library header.

7.670 typeindex File Reference

Classes

- struct [`std::hash< type_index >`](#)
- struct [`std::type_index`](#)

Namespaces

- namespace [`std`](#)

Macros

- `#define _GLIBCXX_TYPEINDEX`

7.670.1 Detailed Description

This is a Standard C++ Library header.

7.671 unordered_map File Reference

Classes

- class `std::__debug::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>`
- class `std::__debug::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>`

Namespaces

- namespace `std`
- namespace `std::__debug`

Macros

- `#define _GLIBCXX_DEBUG_UNORDERED_MAP`

Functions

- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>`
`bool std::__debug::operator==(const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>`
`bool std::__debug::operator==(const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>`
`void std::__debug::swap(unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>`
`void std::__debug::swap(unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_map(_InputIterator, _InputIterator, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, hash<__iter_key_t<_InputIterator>>, equal_to<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_map(_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, hash<__iter_key_t<_InputIterator>>, equal_to<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_map(_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type, _Hash, _Allocator) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Hash, equal_to<__iter_key_t<_InputIterator>>, _Allocator>`
- `template<typename _InputIterator, typename _Hash = hash<__iter_key_t<_InputIterator>>, typename _Pred = equal_to<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_val_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_map(_InputIterator, _InputIterator, typename unordered_map<int, int>::size_type, type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map<__iter_key_t<_InputIterator>, __iter_val_t<_InputIterator>, _Hash, _Pred, _Allocator>`
- `template<typename _Key, typename _Allocator, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_map(initializer_list<pair<_Key, _Tp>>, _Allocator) -> unordered_map<_Key, _Tp, hash<_Key>, equal_to<_Key>, _Allocator>`

- template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type, _Allocator) -> unordered_map< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >
- template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type, _Hash, _Allocator) -> unordered_map< _Key, _Tp, _Hash, equal_to< _Key >, _Allocator >
- template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_map (initializer_list< pair< _Key, _Tp > >, typename unordered_map< int, int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_map< _Key, _Tp, _Hash, _Pred, _Allocator >
- template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (_InputIterator, _InputIterator, _Allocator) -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >
- template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type, _Allocator) -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, hash< __iter_key_t< _InputIterator > >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >
- template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type, _Hash, _Allocator) -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, __iter_key_t< _InputIterator >, equal_to< __iter_key_t< _InputIterator > >, _Allocator >
- template<typename _InputIterator, typename _Hash = hash<__iter_key_t<_InputIterator>>, typename _Pred = equal_to<__iter_key_t<_InputIterator>>, typename _Allocator = allocator<__iter_to_alloc_t<_InputIterator>>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (_InputIterator, _InputIterator, unordered_multimap< int, int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multimap< __iter_key_t< _InputIterator >, __iter_val_t< _InputIterator >, _Hash, _Pred, _Allocator >
- template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (initializer_list< pair< _Key, _Tp > >, _Allocator) -> unordered_multimap< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >
- template<typename _Key, typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type, _Allocator) -> unordered_multimap< _Key, _Tp, hash< _Key >, equal_to< _Key >, _Allocator >
- template<typename _Key, typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type, _Hash, _Allocator) -> unordered_multimap< _Key, _Tp, _Hash, equal_to< _Key >, _Allocator >
- template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>, typename _Allocator = allocator<pair<const _Key, _Tp>>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>
std::__debug::unordered_multimap (initializer_list< pair< _Key, _Tp > >, unordered_multimap< int, int >::size_type={}, _Hash=_Hash(), _Pred=_Pred(), _Allocator=_Allocator()) -> unordered_multimap< _Key, _Tp, _Hash, _Pred, _Allocator >

7.671.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.672 unordered_map File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UNORDERED_MAP`

Typedefs

- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>
using std::experimental::fundamentals_v2::pmr::unordered_map = std::unordered_map<_Key, _Tp, _Hash,
_Pred, polymorphic_allocator< pair< const _Key, _Tp > > >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>
using std::experimental::fundamentals_v2::pmr::unordered_multimap = std::unordered_multimap<_Key, ↵
_Tp, _Hash, _Pred, polymorphic_allocator< pair< const _Key, _Tp > > >`

Functions

- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >
void std::experimental::erase_if (unordered_map< _Key, _Tp, _Hash, _CPred, _Alloc > &__cont, _Predicate
__pred)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >
void std::experimental::erase_if (unordered_multimap< _Key, _Tp, _Hash, _CPred, _Alloc > &__cont, ↵
Predicate __pred)`

7.672.1 Detailed Description

This is a TS C++ Library header.

7.673 unordered_map File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_UNORDERED_MAP`

Typedefs

- `template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>>
using std::pmr::unordered_map = std::unordered_map<_Key, _Tp, _Hash, _Pred, polymorphic_allocator<
pair< const _Key, _Tp > > >`
- `template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>>
using std::pmr::unordered_multimap = std::unordered_multimap<_Key, _Tp, _Hash, _Pred, polymorphic_↵
allocator< pair< const _Key, _Tp > > >`

Functions

- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate>`
`unordered_map< _Key, _Tp, _Hash, _CPred, _Alloc >::size_type std::erase_if (unordered_map< _Key, _Tp,`
`_Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate>`
`unordered_multimap< _Key, _Tp, _Hash, _CPred, _Alloc >::size_type std::erase_if (unordered_multimap< ↵`
`_Key, _Tp, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`

7.673.1 Detailed Description

This is a Standard C++ Library header.

7.674 unordered_set File Reference

Classes

- class `std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`
- class `std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc >`

Namespaces

- namespace `std`
- namespace `std::__debug`

Macros

- `#define _GLIBCXX_DEBUG_UNORDERED_SET`

Functions

- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>`
`bool std::__debug::operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const`
`unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>`
`bool std::__debug::operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const`
`unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>`
`void std::__debug::swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset<`
`_Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>`
`void std::__debug::swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, ↵`
`_Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _Require↵`
`_Allocator<_Allocator>>`
`std::__debug::unordered_multiset (_InputIterator, _InputIterator, unordered_multiset< int >::size_type, ↵`
`_Allocator) -> unordered_multiset< typename iterator_traits< _InputIterator >::value_type, hash< typename`
`iterator_traits< _InputIterator >::value_type >, equal_to< typename iterator_traits< _InputIterator >::value↵`
`_type >, _Allocator >`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename =`
`_RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::unordered_multiset (_InputIterator, _InputIterator, unordered_multiset< int >::size_type, ↵`
`_Hash, _Allocator) -> unordered_multiset< typename iterator_traits< _InputIterator >::value_type, _Hash,`
`equal_to< typename iterator_traits< _InputIterator >::value_type >, _Allocator >`

- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_multiset (`_InputIterator`, `_InputIterator`, `unordered_multiset< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_multiset< typename iterator_traits<_InputIterator>::value_type, _Hash, _Pred, _Allocator >`
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_multiset (`initializer_list< _Tp >`, `unordered_multiset< int >::size_type, _Allocator`) -> `unordered_multiset< _Tp, hash< _Tp >, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_multiset (`initializer_list< _Tp >`, `unordered_multiset< int >::size_type, _Hash, _Allocator`) -> `unordered_multiset< _Tp, _Hash, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_multiset (`initializer_list< _Tp >`, `unordered_multiset< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_multiset< _Tp, _Hash, _Pred, _Allocator >`
- `template<typename _InputIterator, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`_InputIterator`, `_InputIterator`, `unordered_set< int >::size_type, _Allocator`) -> `unordered_set< typename iterator_traits<_InputIterator>::value_type, hash< typename iterator_traits<_InputIterator>::value_type>, equal_to< typename iterator_traits<_InputIterator>::value_type>, _Allocator >`
- `template<typename _InputIterator, typename _Hash, typename _Allocator, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`_InputIterator`, `_InputIterator`, `unordered_set< int >::size_type, _Hash, _Allocator`) -> `unordered_set< typename iterator_traits<_InputIterator>::value_type, _Hash, equal_to< typename iterator_traits<_InputIterator>::value_type>, _Allocator >`
- `template<typename _InputIterator, typename _Hash = hash<typename iterator_traits<_InputIterator>::value_type>, typename _Pred = equal_to<typename iterator_traits<_InputIterator>::value_type>, typename _Allocator = allocator<typename iterator_traits<_InputIterator>::value_type>, typename = _RequireInputIter<_InputIterator>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`_InputIterator`, `_InputIterator`, `unordered_set< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_set< typename iterator_traits<_InputIterator>::value_type, _Hash, _Pred, _Allocator >`
- `template<typename _Tp, typename _Allocator, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`initializer_list< _Tp >`, `unordered_set< int >::size_type, _Allocator`) -> `unordered_set< _Tp, hash< _Tp >, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash, typename _Allocator, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`initializer_list< _Tp >`, `unordered_set< int >::size_type, _Hash, _Allocator`) -> `unordered_set< _Tp, _Hash, equal_to< _Tp >, _Allocator >`
- `template<typename _Tp, typename _Hash = hash<_Tp>, typename _Pred = equal_to<_Tp>, typename _Allocator = allocator<_Tp>, typename = _RequireNotAllocatorOrIntegral<_Hash>, typename = _RequireNotAllocator<_Pred>, typename = _RequireAllocator<_Allocator>>>`
std::__debug::unordered_set (`initializer_list< _Tp >`, `unordered_set< int >::size_type={}`, `_Hash=_Hash()`, `_Pred=_Pred()`, `_Allocator=_Allocator()`) -> `unordered_set< _Tp, _Hash, _Pred, _Allocator >`

7.674.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.675 unordered_set File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UNORDERED_SET`

Typedefs

- `template<typename _Key, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>`
using `std::experimental::fundamentals_v2::pmr::unordered_multiset` = `std::unordered_multiset<_Key, _Hash, _Pred, polymorphic_allocator<_Key>>`
- `template<typename _Key, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>`
using `std::experimental::fundamentals_v2::pmr::unordered_set` = `std::unordered_set<_Key, _Hash, _Pred, polymorphic_allocator<_Key>>`

Functions

- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
void `std::experimental::erase_if` (`unordered_multiset<_Key, _Hash, _CPred, _Alloc> &__cont`, `_Predicate __pred`)
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
void `std::experimental::erase_if` (`unordered_set<_Key, _Hash, _CPred, _Alloc> &__cont`, `_Predicate __pred`)

7.675.1 Detailed Description

This is a TS C++ Library header.

7.676 unordered_set File Reference

Namespaces

- namespace [std](#)

Macros

- `#define _GLIBCXX_UNORDERED_SET`

Typedefs

- `template<typename _Key, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>>>`
using `std::pmr::unordered_multiset` = `std::unordered_multiset<_Key, _Hash, _Pred, polymorphic_allocator<_Key>>`
- `template<typename _Key, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>>>`
using `std::pmr::unordered_set` = `std::unordered_set<_Key, _Hash, _Pred, polymorphic_allocator<_Key>>`

Functions

- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
`unordered_multiset<_Key, _Hash, _CPred, _Alloc>::size_type` `std::erase_if` (`unordered_multiset<_Key, _Hash, _CPred, _Alloc> &__cont`, `_Predicate __pred`)

- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >
unordered_set< _Key, _Hash, _CPred, _Alloc >::size_type std::erase_if (unordered_set< _Key, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)`

7.676.1 Detailed Description

This is a Standard C++ Library header.

7.677 utility File Reference

Namespaces

- namespace `std`
- namespace `std::experimental`

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UTILITY`

Typedefs

- using `std::experimental::erased_type` = `std::__erased_type`

7.677.1 Detailed Description

This is a TS C++ Library header.

7.678 utility File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_as_const`
- `#define __cpp_lib_exchange_function`
- `#define __cpp_lib_integer_comparison_functions`
- `#define _GLIBCXX_UTILITY`

Functions

- `template<typename _Tp >
constexpr add_const_t< _Tp > & std::as_const (_Tp &__t) noexcept`
- `template<typename _Tp >
void std::as_const (const _Tp &&)=delete`
- `template<typename _Tp, typename _Up >
constexpr bool std::cmp_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >
constexpr bool std::cmp_greater (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >
constexpr bool std::cmp_greater_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >
constexpr bool std::cmp_less (_Tp __t, _Up __u) noexcept`

- `template<typename _Tp, typename _Up >`
`constexpr bool std::cmp_less_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::cmp_not_equal (_Tp __t, _Up __u) noexcept`
- `template<typename _Tp, typename _Up = _Tp>`
`constexpr _Tp std::exchange (_Tp &__obj, _Up &&__new_val) noexcept(__and_< is_nothrow_move_constructible<_Tp>, is_nothrow_assignable<_Tp &, _Up> >::value)`
- `template<typename _Up, typename _Tp >`
`constexpr bool std::in_range (_Tp __t) noexcept`

7.678.1 Detailed Description

This is a Standard C++ Library header.

7.679 valarray File Reference

Classes

- class [std::valarray<_Tp>](#)

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define _GLIBCXX_VALARRAY`

Functions

- `template<class _Tp >`
`const _Tp * std::begin (const valarray<_Tp> &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::begin (valarray<_Tp> &__va) noexcept`
- `template<class _Tp >`
`const _Tp * std::end (const valarray<_Tp> &__va) noexcept`
- `template<class _Tp >`
`_Tp * std::end (valarray<_Tp> &__va) noexcept`
- `template<typename _Tp, size_t _Nm>`
`std::valarray (const _Tp(&)[_Nm], size_t) -> valarray<_Tp>`

7.679.1 Detailed Description

This is a Standard C++ Library header.

7.680 variant File Reference

Namespaces

- namespace [std](#)
- namespace [std::__detail](#)

Macros

- `#define __cpp_lib_variant`
- `#define _GLIBCXX_VARIANT`
- `#define _VARIANT_RELATION_FUNCTION_TEMPLATE(__OP)`
- `#define _VARIANT_RELATION_FUNCTION_TEMPLATE(__OP, __NAME)`

Typedefs

- `template<size_t _Np, typename _Variant, typename _AsV = decltype(__variant::__as(std::declval<_Variant>())), typename _Tp = variant<_alternative_t<_Np, remove_reference_t<_AsV>>>>`
`using std::__detail::__variant::__get_t = __conditional_t< is_lvalue_reference_v<_Variant>, _Tp &, _Tp &&>`
- `using std::__detail::__variant::__index_type = __select_index<_Types...>`
- `template<typename _Visitor, typename... _Variants>`
`using std::__detail::__variant::__visit_result_t = invoke_result_t<_Visitor, __get_t<0, _Variants>...>`
- `template<typename... _Types>`
`using std::__detail::__variant::__Copy_assign_alias = _Copy_assign_base<_Traits<_Types...>::__S_trivial_copy_assign, _Types...>`
- `template<typename... _Types>`
`using std::__detail::__variant::__Copy_ctor_alias = _Copy_ctor_base<_Traits<_Types...>::__S_trivial_copy_ctor, _Types...>`
- `template<typename _Tp, typename _Variant>`
`using std::__detail::__variant::__FUN_type = decltype(_Build_FUNs<_Tp, _Variant>::__S_fun(std::declval<_Tp>()))`
- `template<typename... _Types>`
`using std::__detail::__variant::__Move_assign_alias = _Move_assign_base<_Traits<_Types...>::__S_trivial_move_assign, _Types...>`
- `template<typename... _Types>`
`using std::__detail::__variant::__Move_ctor_alias = _Move_ctor_base<_Traits<_Types...>::__S_trivial_move_ctor, _Types...>`
- `template<typename... _Types>`
`using std::__detail::__variant::__Variant_storage_alias = _Variant_storage<_Traits<_Types...>::__S_trivial_dtor, _Types...>`
- `template<size_t _Np, typename _Variant>`
`using std::variant_alternative_t = typename variant_alternative<_Np, _Variant>::type`

Functions

- `template<typename... _Types>`
`constexpr const std::variant<_Types...> && std::__detail::__variant::__as (const std::variant<_Types...> &&__v) noexcept`
- `template<typename... _Types>`
`constexpr const std::variant<_Types...> & std::__detail::__variant::__as (const std::variant<_Types...> &__v) noexcept`
- `template<typename... _Types>`
`constexpr std::variant<_Types...> && std::__detail::__variant::__as (std::variant<_Types...> &&__v) noexcept`
- `template<typename... _Types>`
`constexpr std::variant<_Types...> & std::__detail::__variant::__as (std::variant<_Types...> &__v) noexcept`
- `template<typename _Visitor, typename _Variant, size_t... _Idxs>`
`constexpr bool std::__detail::__variant::__check_visitor_results (std::index_sequence<_Idxs...>)`
- `template<typename _Result_type, typename _Visitor, typename... _Variants>`
`constexpr decltype(auto) std::__do_visit (_Visitor &&__visitor, _Variants &&... __variants)`

- `template<size_t _Np, bool _Triv, typename... _Types, typename... _Args>`
`constexpr void std::detail::variant::emplace (_Variant_storage< _Triv, _Types... > &__v, _Args &&... __args)`
- `template<size_t _Np, typename _Variant >`
`constexpr decltype(auto) std::detail::variant::get (_Variant &&__v) noexcept`
- `template<size_t _Np, typename _Union >`
`constexpr decltype(auto) std::detail::variant::get_n (_Union &&__u) noexcept`
- `template<typename _Visitor, typename... _Variants>`
`constexpr void std::detail::variant::raw_idx_visit (_Visitor &&__visitor, _Variants &&... __variants)`
- `template<typename _Visitor, typename... _Variants>`
`constexpr void std::detail::variant::raw_visit (_Visitor &&__visitor, _Variants &&... __variants)`
- `void std::throw_bad_variant_access (bool __valueless)`
- `void std::throw_bad_variant_access (const char * __what)`
- `template<typename... _Types, typename _Tp >`
`constexpr decltype(auto) std::variant_cast (_Tp &&__rhs)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > const && std::get (const variant< _Types... > &&)`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp && std::get (const variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr const variant_alternative_t< _Np, variant< _Types... > > && std::get (const variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > const & std::get (const variant< _Types... > &)`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & std::get (const variant< _Types... > &__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr const variant_alternative_t< _Np, variant< _Types... > > & std::get (const variant< _Types... > &__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > && std::get (variant< _Types... > &&)`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && std::get (variant< _Types... > &&__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr variant_alternative_t< _Np, variant< _Types... > > & std::get (variant< _Types... > &)`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & std::get (variant< _Types... > &__v)`
- `template<size_t _Np, typename... _Types>`
`constexpr add_pointer_t< const variant_alternative_t< _Np, variant< _Types... > > > std::get_if (const variant< _Types... > *__ptr) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr add_pointer_t< const _Tp > std::get_if (const variant< _Types... > *__ptr) noexcept`
- `template<size_t _Np, typename... _Types>`
`constexpr add_pointer_t< variant_alternative_t< _Np, variant< _Types... > > > std::get_if (variant< _Types... > *__ptr) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr add_pointer_t< _Tp > std::get_if (variant< _Types... > *__ptr) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr bool std::holds_alternative (const variant< _Types... > &__v) noexcept`
- `template<typename... _Types>`
`constexpr bool std::operator!= (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)`

- constexpr bool **std::operator!=** (monostate, monostate) noexcept
- template<typename... _Types>
constexpr bool **std::operator<** (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)
- constexpr bool **std::operator<** (monostate, monostate) noexcept
- template<typename... _Types>
constexpr bool **std::operator<=** (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)
- constexpr bool **std::operator<=** (monostate, monostate) noexcept
- template<typename... _Types>
constexpr bool **std::operator==** (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)
- constexpr bool **std::operator==** (monostate, monostate) noexcept
- template<typename... _Types>
constexpr bool **std::operator>** (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)
- constexpr bool **std::operator>** (monostate, monostate) noexcept
- template<typename... _Types>
constexpr bool **std::operator>=** (const variant< _Types... > &__lhs, const variant< _Types... > &__rhs)
- constexpr bool **std::operator>=** (monostate, monostate) noexcept
- template<typename... _Types>
[enable_if_t](#)<!((is_move_constructible_v< _Types > &&...) &&(is_swappable_v< _Types > &&...))> **std::swap**
(variant< _Types... > &, variant< _Types... > &)=delete
- template<typename... _Types>
constexpr [enable_if_t](#)<(is_move_constructible_v< _Types > &&...) &&(is_swappable_v< _Types > &&...)>
std::swap (variant< _Types... > &__lhs, variant< _Types... > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))
- template<typename _Visitor, typename... _Variants>
constexpr __detail::__variant::__visit_result_t< _Visitor, _Variants... > **std::visit** (_Visitor &&, _Variants &&...)
- template<typename _Visitor, typename... _Variants>
constexpr __detail::__variant::__visit_result_t< _Visitor, _Variants... > **std::visit** (_Visitor &&__visitor, _Variants
&&... __variants)
- template<typename _Res, typename _Visitor, typename... _Variants>
constexpr _Res **std::visit** (_Visitor &&__visitor, _Variants &&... __variants)

Variables

- template<typename _Tp, typename... _Types>
constexpr bool **std::__detail::__variant::__exactly_once**
- template<typename _Tp, typename... _Types>
constexpr bool **std::__detail::__variant::__same_types**
- __index_type **std::__detail::__variant::__M_index**
- template<typename _Tp >
__Variadic_union< _Types... > **std::__detail::__variant::__M_u**
- constexpr size_t **std::variant_npos**
- template<typename _Variant >
constexpr size_t **std::variant_size_v**
- template<typename... _Types>
constexpr size_t **std::variant_size_v**< const variant< _Types... > >
- template<typename... _Types>
constexpr size_t **std::variant_size_v**< variant< _Types... > >

7.680.1 Detailed Description

This is the <variant> C++ Library header.

7.681 vector File Reference

Classes

- class [__gnu_debug::Safe_vector<_SafeSequence, _BaseSequence>](#)
- struct [std::hash<__debug::vector<bool, _Alloc>>](#)
- class [std::__debug::vector<_Tp, _Allocator>](#)

Namespaces

- namespace [__gnu_debug](#)
- namespace [std](#)
- namespace [std::__debug](#)
- namespace [std::__detail](#)

Macros

- `#define _GLIBCXX_DEBUG_VECTOR`

Functions

- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator!= (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator< (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator<= (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator== (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator> (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator>= (const vector<_Tp, _Alloc> &__lhs, const vector<_Tp, _Alloc> &__rhs)`
- `template<typename _Tp, typename _Alloc>`
`void std::__debug::swap (vector<_Tp, _Alloc> &__lhs, vector<_Tp, _Alloc> &__rhs) noexcept(/*conditional */)`
- `template<typename _InputIterator, typename _ValT = typename iterator_traits<_InputIterator>::value_type, typename _Allocator = allocator<_ValT>, typename = _RequireInputIter<_InputIterator>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::vector (_InputIterator, _InputIterator, _Allocator=_Allocator()) -> vector<_ValT, _Allocator>`
- `template<typename _Tp, typename _Allocator = allocator<_Tp>, typename = _RequireAllocator<_Allocator>>`
`std::__debug::vector (size_t, _Tp, _Allocator=_Allocator()) -> vector<_Tp, _Allocator>`

7.681.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

7.682 vector File Reference

Namespaces

- namespace [std](#)
- namespace [std::experimental](#)

Macros

- `#define __cpp_lib_experimental_erase_if`
- `#define _GLIBCXX_EXPERIMENTAL_VECTOR`

Typedefs

- `template<typename _Tp >`
using `std::experimental::fundamentals_v2::pmr::vector` = `std::vector`< _Tp, `polymorphic_allocator`< _Tp >
>

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
void `std::experimental::erase` (`vector`< _Tp, _Alloc > &__cont, const _Up &__value)
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
void `std::experimental::erase_if` (`vector`< _Tp, _Alloc > &__cont, _Predicate __pred)

7.682.1 Detailed Description

This is a TS C++ Library header.

7.683 vector File Reference

Namespaces

- namespace `std`

Macros

- `#define __cpp_lib_erase_if`
- `#define _GLIBCXX_VECTOR`

Typedefs

- `template<typename _Tp >`
using `std::pmr::vector` = `std::vector`< _Tp, `polymorphic_allocator`< _Tp > >

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
constexpr `vector`< _Tp, _Alloc >::size_type `std::erase` (`vector`< _Tp, _Alloc > &__cont, const _Up &__value)
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
constexpr `vector`< _Tp, _Alloc >::size_type `std::erase_if` (`vector`< _Tp, _Alloc > &__cont, _Predicate __pred)

7.683.1 Detailed Description

This is a Standard C++ Library header.

7.684 atomic_word.h File Reference

Macros

- `#define _GLIBCXX_READ_MEM_BARRIER`
- `#define _GLIBCXX_WRITE_MEM_BARRIER`

Typedefs

- typedef int `_Atomic_word`

7.684.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

7.685 `basic_file.h` File Reference

Namespaces

- namespace `std`

7.685.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

7.686 `c++allocator.h` File Reference

Namespaces

- namespace `std`

Typedefs

- template<typename `_Tp` >
using `std::__allocator_base` = `__new_allocator`< `_Tp` >

7.686.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

7.687 `c++config.h` File Reference

Namespaces

- namespace `__gnu_cxx`
- namespace `std`

Macros

- `#define __GLIBCXX__`
- `#define __glibcxx_assert(cond)`
- `#define __glibcxx_constexpr_assert(cond)`
- `#define __N(msgid)`
- `#define _GLIBCXX11_DEPRECATED`
- `#define _GLIBCXX11_DEPRECATED_SUGGEST(ALT)`
- `#define _GLIBCXX11_USE_C99_MATH`
- `#define _GLIBCXX11_USE_C99_STDIO`
- `#define _GLIBCXX11_USE_C99_STDLIB`
- `#define _GLIBCXX11_USE_C99_WCHAR`
- `#define _GLIBCXX14_DEPRECATED`
- `#define _GLIBCXX14_DEPRECATED_SUGGEST(ALT)`

- `#define _GLIBCXX17_DEPRECATED`
- `#define _GLIBCXX17_DEPRECATED_SUGGEST(ALT)`
- `#define _GLIBCXX20_DEPRECATED(MSG)`
- `#define _GLIBCXX20_DEPRECATED_SUGGEST(ALT)`
- `#define _GLIBCXX23_CONSTEXPR`
- `#define _GLIBCXX98_USE_C99_COMPLEX`
- `#define _GLIBCXX98_USE_C99_MATH`
- `#define _GLIBCXX98_USE_C99_STDIO`
- `#define _GLIBCXX98_USE_C99_STDLIB`
- `#define _GLIBCXX98_USE_C99_WCHAR`
- `#define _GLIBCXX_ABI_TAG_CXX11`
- `#define _GLIBCXX_ATOMIC_BUILTINS`
- `#define _GLIBCXX_BEGIN_EXTERN_C`
- `#define _GLIBCXX_BEGIN_NAMESPACE_ALGO`
- `#define _GLIBCXX_BEGIN_NAMESPACE_CONTAINER`
- `#define _GLIBCXX_BEGIN_NAMESPACE_CXX11`
- `#define _GLIBCXX_BEGIN_NAMESPACE_LDBL`
- `#define _GLIBCXX_BEGIN_NAMESPACE_LDBL_OR_CXX11`
- `#define _GLIBCXX_BEGIN_NAMESPACE_VERSION`
- `#define _GLIBCXX_DARWIN_USE_64_BIT_INODE`
- `#define _GLIBCXX_DEFAULT_ABI_TAG`
- `#define _GLIBCXX_DEPRECATED`
- `#define _GLIBCXX_DEPRECATED_SUGGEST(ALT)`
- `#define _GLIBCXX_END_EXTERN_C`
- `#define _GLIBCXX_END_NAMESPACE_ALGO`
- `#define _GLIBCXX_END_NAMESPACE_CONTAINER`
- `#define _GLIBCXX_END_NAMESPACE_CXX11`
- `#define _GLIBCXX_END_NAMESPACE_LDBL`
- `#define _GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11`
- `#define _GLIBCXX_END_NAMESPACE_VERSION`
- `#define _GLIBCXX_EXTERN_TEMPLATE`
- `#define _GLIBCXX_FAST_MATH`
- `#define _GLIBCXX_FULLY_DYNAMIC_STRING`
- `#define _GLIBCXX_HAS_BUILTIN(B)`
- `#define _GLIBCXX_HAVE__CXA_THREAD_ATEXIT_IMPL`
- `#define _GLIBCXX_HAVE_ACOSF`
- `#define _GLIBCXX_HAVE_ACOSL`
- `#define _GLIBCXX_HAVE_ALIGNED_ALLOC`
- `#define _GLIBCXX_HAVE_ARC4RANDOM`
- `#define _GLIBCXX_HAVE_ARPA_INET_H`
- `#define _GLIBCXX_HAVE_AS_SYMVER_DIRECTIVE`
- `#define _GLIBCXX_HAVE_ASINF`
- `#define _GLIBCXX_HAVE_ASINL`
- `#define _GLIBCXX_HAVE_AT_QUICK_EXIT`
- `#define _GLIBCXX_HAVE_ATAN2F`
- `#define _GLIBCXX_HAVE_ATAN2L`
- `#define _GLIBCXX_HAVE_ATANF`
- `#define _GLIBCXX_HAVE_ATANL`
- `#define _GLIBCXX_HAVE_ATOMIC_LOCK_POLICY`
- `#define _GLIBCXX_HAVE_ATTRIBUTE_VISIBILITY`
- `#define _GLIBCXX_HAVE_BUILTIN_HAS_UNIQ_OBJ_REP`

- #define `_GLIBCXX_HAVE_BUILTIN_IS_AGGREGATE`
- #define `_GLIBCXX_HAVE_BUILTIN_IS_SAME`
- #define `_GLIBCXX_HAVE_BUILTIN_LAUNDER`
- #define `_GLIBCXX_HAVE_CEILF`
- #define `_GLIBCXX_HAVE_CEILL`
- #define `_GLIBCXX_HAVE_COMPLEX_H`
- #define `_GLIBCXX_HAVE_COSF`
- #define `_GLIBCXX_HAVE_COSHF`
- #define `_GLIBCXX_HAVE_COSHL`
- #define `_GLIBCXX_HAVE_COSL`
- #define `_GLIBCXX_HAVE_DECL_STRNLEN`
- #define `_GLIBCXX_HAVE_DIRENT_H`
- #define `_GLIBCXX_HAVE_DIRFD`
- #define `_GLIBCXX_HAVE_DLFCN_H`
- #define `_GLIBCXX_HAVE_ENDIAN_H`
- #define `_GLIBCXX_HAVE_EXCEPTION_PTR_SINCE_GCC46`
- #define `_GLIBCXX_HAVE_EXECINFO_H`
- #define `_GLIBCXX_HAVE_EXPF`
- #define `_GLIBCXX_HAVE_EXPL`
- #define `_GLIBCXX_HAVE_FABSF`
- #define `_GLIBCXX_HAVE_FABSL`
- #define `_GLIBCXX_HAVE_FCNTL_H`
- #define `_GLIBCXX_HAVE_FDOPENDIR`
- #define `_GLIBCXX_HAVE_FENV_H`
- #define `_GLIBCXX_HAVE_FINITE`
- #define `_GLIBCXX_HAVE_FINITEF`
- #define `_GLIBCXX_HAVE_FINITEL`
- #define `_GLIBCXX_HAVE_FLOAT_H`
- #define `_GLIBCXX_HAVE_FLOORF`
- #define `_GLIBCXX_HAVE_FLOORL`
- #define `_GLIBCXX_HAVE_FMODF`
- #define `_GLIBCXX_HAVE_FMODL`
- #define `_GLIBCXX_HAVE_FREXPF`
- #define `_GLIBCXX_HAVE_FREXPL`
- #define `_GLIBCXX_HAVE_GETENTROPY`
- #define `_GLIBCXX_HAVE_GETIPINFO`
- #define `_GLIBCXX_HAVE_GETS`
- #define `_GLIBCXX_HAVE_HYPOT`
- #define `_GLIBCXX_HAVE_HYPOTF`
- #define `_GLIBCXX_HAVE_HYPOTL`
- #define `_GLIBCXX_HAVE_ICONv`
- #define `_GLIBCXX_HAVE_INTTYPES_H`
- #define `_GLIBCXX_HAVE_IS_CONSTANT_EVALUATED`
- #define `_GLIBCXX_HAVE_ISINF`
- #define `_GLIBCXX_HAVE_ISINFF`
- #define `_GLIBCXX_HAVE_ISINFL`
- #define `_GLIBCXX_HAVE_ISNAN`
- #define `_GLIBCXX_HAVE_ISNANF`
- #define `_GLIBCXX_HAVE_ISNANL`
- #define `_GLIBCXX_HAVE_ISWBLANK`
- #define `_GLIBCXX_HAVE_LC_MESSAGES`

- `#define _GLIBCXX_HAVE_LDEXPF`
- `#define _GLIBCXX_HAVE_LDEXPL`
- `#define _GLIBCXX_HAVE_LIBINTL_H`
- `#define _GLIBCXX_HAVE_LIMIT_AS`
- `#define _GLIBCXX_HAVE_LIMIT_DATA`
- `#define _GLIBCXX_HAVE_LIMIT_FSIZE`
- `#define _GLIBCXX_HAVE_LIMIT_RSS`
- `#define _GLIBCXX_HAVE_LIMIT_VMEM`
- `#define _GLIBCXX_HAVE_LINK`
- `#define _GLIBCXX_HAVE_LINK_H`
- `#define _GLIBCXX_HAVE_LINUX_FUTEX`
- `#define _GLIBCXX_HAVE_LINUX_RANDOM_H`
- `#define _GLIBCXX_HAVE_LINUX_TYPES_H`
- `#define _GLIBCXX_HAVE_LOCALE_H`
- `#define _GLIBCXX_HAVE_LOG10F`
- `#define _GLIBCXX_HAVE_LOG10L`
- `#define _GLIBCXX_HAVE_LOGF`
- `#define _GLIBCXX_HAVE_LOGL`
- `#define _GLIBCXX_HAVE_MBSTATE_T`
- `#define _GLIBCXX_HAVE_MEMALIGN`
- `#define _GLIBCXX_HAVE_MEMORY_H`
- `#define _GLIBCXX_HAVE_MODFF`
- `#define _GLIBCXX_HAVE_MODFL`
- `#define _GLIBCXX_HAVE_NETDB_H`
- `#define _GLIBCXX_HAVE_NETINET_IN_H`
- `#define _GLIBCXX_HAVE_NETINET_TCP_H`
- `#define _GLIBCXX_HAVE_OPENAT`
- `#define _GLIBCXX_HAVE_POLL`
- `#define _GLIBCXX_HAVE_POLL_H`
- `#define _GLIBCXX_HAVE_POSIX_MEMALIGN`
- `#define _GLIBCXX_HAVE_POSIX_SEMAPHORE`
- `#define _GLIBCXX_HAVE_POWF`
- `#define _GLIBCXX_HAVE_POWL`
- `#define _GLIBCXX_HAVE_QUICK_EXIT`
- `#define _GLIBCXX_HAVE_READLINK`
- `#define _GLIBCXX_HAVE_S_ISREG`
- `#define _GLIBCXX_HAVE_SECURE_GETENV`
- `#define _GLIBCXX_HAVE_SETENV`
- `#define _GLIBCXX_HAVE_SINCOS`
- `#define _GLIBCXX_HAVE_SINCOSF`
- `#define _GLIBCXX_HAVE_SINCOSL`
- `#define _GLIBCXX_HAVE_SINF`
- `#define _GLIBCXX_HAVE_SINHF`
- `#define _GLIBCXX_HAVE_SINHL`
- `#define _GLIBCXX_HAVE_SINL`
- `#define _GLIBCXX_HAVE_SOCKETATMARK`
- `#define _GLIBCXX_HAVE_SQRTF`
- `#define _GLIBCXX_HAVE_SQRTL`
- `#define _GLIBCXX_HAVE_STDALIGN_H`
- `#define _GLIBCXX_HAVE_STDBOOL_H`
- `#define _GLIBCXX_HAVE_STDINT_H`

- #define _GLIBCXX_HAVE_STDLIB_H
- #define _GLIBCXX_HAVE_STRERROR_L
- #define _GLIBCXX_HAVE_STRERROR_R
- #define _GLIBCXX_HAVE_STRING_H
- #define _GLIBCXX_HAVE_STRINGS_H
- #define _GLIBCXX_HAVE_STRTOF
- #define _GLIBCXX_HAVE_STRTOLD
- #define _GLIBCXX_HAVE_STRUCT_DIRENT_D_TYPE
- #define _GLIBCXX_HAVE_STRXFRM_L
- #define _GLIBCXX_HAVE_SYMLINK
- #define _GLIBCXX_HAVE_SYMVER_SYMBOL_RENAMING_RUNTIME_SUPPORT
- #define _GLIBCXX_HAVE_SYS_IOCTL_H
- #define _GLIBCXX_HAVE_SYS_IPC_H
- #define _GLIBCXX_HAVE_SYS_PARAM_H
- #define _GLIBCXX_HAVE_SYS_RESOURCE_H
- #define _GLIBCXX_HAVE_SYS_SDT_H
- #define _GLIBCXX_HAVE_SYS_SEM_H
- #define _GLIBCXX_HAVE_SYS_SOCKET_H
- #define _GLIBCXX_HAVE_SYS_STAT_H
- #define _GLIBCXX_HAVE_SYS_STATVFS_H
- #define _GLIBCXX_HAVE_SYS_SYSINFO_H
- #define _GLIBCXX_HAVE_SYS_TIME_H
- #define _GLIBCXX_HAVE_SYS_TYPES_H
- #define _GLIBCXX_HAVE_SYS_UIO_H
- #define _GLIBCXX_HAVE_TANF
- #define _GLIBCXX_HAVE_TANHF
- #define _GLIBCXX_HAVE_TANHL
- #define _GLIBCXX_HAVE_TANL
- #define _GLIBCXX_HAVE_TGMATH_H
- #define _GLIBCXX_HAVE_TIMESPEC_GET
- #define _GLIBCXX_HAVE_TLS
- #define _GLIBCXX_HAVE_TRUNCATE
- #define _GLIBCXX_HAVE_UCHAR_H
- #define _GLIBCXX_HAVE_UNISTD_H
- #define _GLIBCXX_HAVE_UNLINKAT
- #define _GLIBCXX_HAVE_USELOCALE
- #define _GLIBCXX_HAVE_UTIME_H
- #define _GLIBCXX_HAVE_VFWSCANF
- #define _GLIBCXX_HAVE_VSWSCANF
- #define _GLIBCXX_HAVE_VWSCANF
- #define _GLIBCXX_HAVE_WCHAR_H
- #define _GLIBCXX_HAVE_WCSTOF
- #define _GLIBCXX_HAVE_WCTYPE_H
- #define _GLIBCXX_HAVE_WRITEV
- #define _GLIBCXX_HOSTED
- #define _GLIBCXX_ICONV_CONST
- #define _GLIBCXX_INLINE_VERSION
- #define _GLIBCXX_LT_OBJDIR
- #define _GLIBCXX_MANGLE_SIZE_T
- #define _GLIBCXX_NAMESPACE_CXX11
- #define _GLIBCXX_NAMESPACE_LDBL

- #define _GLIBCXX_NAMESPACE_LDBL_OR_CXX11
- #define _GLIBCXX_NODISCARD
- #define _GLIBCXX_NOEXCEPT_PARM
- #define _GLIBCXX_NOEXCEPT_QUAL
- #define _GLIBCXX_PACKAGE__GLIBCXX_VERSION
- #define _GLIBCXX_PACKAGE_BUGREPORT
- #define _GLIBCXX_PACKAGE_NAME
- #define _GLIBCXX_PACKAGE_STRING
- #define _GLIBCXX_PACKAGE_TARNAME
- #define _GLIBCXX_PACKAGE_URL
- #define _GLIBCXX_PSEUDO_VISIBILITY(V)
- #define _GLIBCXX_RELEASE
- #define _GLIBCXX_RES_LIMITS
- #define _GLIBCXX_STD_A
- #define _GLIBCXX_STD_C
- #define _GLIBCXX_STDC_HEADERS
- #define _GLIBCXX_STDIO_EOF
- #define _GLIBCXX_STDIO_SEEK_CUR
- #define _GLIBCXX_STDIO_SEEK_END
- #define _GLIBCXX_SYMVER
- #define _GLIBCXX_SYMVER_GNU
- #define _GLIBCXX_SYNCHRONIZATION_HAPPENS_AFTER(A)
- #define _GLIBCXX_SYNCHRONIZATION_HAPPENS_BEFORE(A)
- #define _GLIBCXX_THROW_OR_ABORT(_EXC)
- #define _GLIBCXX_TXN_SAFE
- #define _GLIBCXX_TXN_SAFE_DYN
- #define _GLIBCXX_USE_ALLOCATOR_NEW
- #define _GLIBCXX_USE_C11_UCHAR_CXX11
- #define _GLIBCXX_USE_C99
- #define _GLIBCXX_USE_C99_COMPLEX
- #define _GLIBCXX_USE_C99_COMPLEX_TR1
- #define _GLIBCXX_USE_C99_CTYPE_TR1
- #define _GLIBCXX_USE_C99_FENV_TR1
- #define _GLIBCXX_USE_C99_INTTYPES_TR1
- #define _GLIBCXX_USE_C99_INTTYPES_WCHAR_T_TR1
- #define _GLIBCXX_USE_C99_MATH
- #define _GLIBCXX_USE_C99_MATH_TR1
- #define _GLIBCXX_USE_C99_STDINT_TR1
- #define _GLIBCXX_USE_C99_STDIO
- #define _GLIBCXX_USE_C99_STDLIB
- #define _GLIBCXX_USE_C99_WCHAR
- #define _GLIBCXX_USE_CLOCK_MONOTONIC
- #define _GLIBCXX_USE_CLOCK_REALTIME
- #define _GLIBCXX_USE_CXX11_ABI
- #define _GLIBCXX_USE_DECIMAL_FLOAT
- #define _GLIBCXX_USE_DEV_RANDOM
- #define _GLIBCXX_USE_DUAL_ABI
- #define _GLIBCXX_USE_FCHMOD
- #define _GLIBCXX_USE_FCHMODAT
- #define _GLIBCXX_USE_GET_NPROCS
- #define _GLIBCXX_USE_GETTIMEOFDAY

- `#define _GLIBCXX_USE_LFS`
- `#define _GLIBCXX_USE_LONG_LONG`
- `#define _GLIBCXX_USE_LSTAT`
- `#define _GLIBCXX_USE_NANOSLEEP`
- `#define _GLIBCXX_USE_NLS`
- `#define _GLIBCXX_USE_PTHREAD_COND_CLOCKWAIT`
- `#define _GLIBCXX_USE_PTHREAD_MUTEX_CLOCKLOCK`
- `#define _GLIBCXX_USE_PTHREAD_RWLOCK_CLOCKLOCK`
- `#define _GLIBCXX_USE_RANDOM_TR1`
- `#define _GLIBCXX_USE_REALPATH`
- `#define _GLIBCXX_USE_SC_NPROCESSORS_ONLN`
- `#define _GLIBCXX_USE_SCHED_YIELD`
- `#define _GLIBCXX_USE_SENDFILE`
- `#define _GLIBCXX_USE_ST_MTIM`
- `#define _GLIBCXX_USE_STD_SPEC_FUNCS`
- `#define _GLIBCXX_USE_TMPNAM`
- `#define _GLIBCXX_USE_UCHAR_C8RTOMB_MBRTOC8_CXX20`
- `#define _GLIBCXX_USE_UCHAR_C8RTOMB_MBRTOC8_FCHAR8_T`
- `#define _GLIBCXX_USE_UTCIME`
- `#define _GLIBCXX_USE_UTIMENSAT`
- `#define _GLIBCXX_USE_WCHAR_T`
- `#define _GLIBCXX_USE_WEAK_REF`
- `#define _GLIBCXX_VERBOSE`
- `#define _GLIBCXX_VERBOSE_ASSERT`
- `#define _GLIBCXX_VISIBILITY(V)`
- `#define _GLIBCXX_WEAK_DEFINITION`
- `#define _GLIBCXX_X86_RDRAND`
- `#define _GLIBCXX_X86_RDSEED`
- `#define _GTHREAD_USE_MUTEX_TIMEDLOCK`

Typedefs

- `typedef decltype(nullptr) std::nullptr_t`
- `typedef __PTRDIFF_TYPE__ std::ptrdiff_t`
- `typedef __SIZE_TYPE__ std::size_t`

Functions

- `constexpr bool std::__is_constant_evaluated () noexcept`
- `void std::__terminate () noexcept`

7.687.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<version>`.

7.688 c++io.h File Reference

Namespaces

- namespace [std](#)

Typedefs

- typedef FILE **std::__c_file**
- typedef __pthread_mutex_t **std::__c_lock**

7.688.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

7.689 c++locale.h File Reference

Namespaces

- namespace [std](#)

Macros

- #define **_GLIBCXX_C_LOCALE_GNU**
- #define **_GLIBCXX_NUM_CATEGORIES**

Typedefs

- typedef __locale_t **std::__c_locale**

Functions

- int **std::__convert_from_v** (const __c_locale &__cloc, char * __out, const int __size, const char * __fmt,...)

7.689.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.690 c++locale_internal.h File Reference

Namespaces

- namespace [std](#)

Functions

- Catalogs & **std::get_catalogs** ()

7.690.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.691 compatibility.h File Reference

Namespaces

- namespace [__gnu_parallel](#)

Functions

- `template<typename _Tp >`
`_Tp __gnu_parallel::__add_omp` (volatile `_Tp *``__ptr`, `_Tp` `__addend`)
- `template<typename _Tp >`
`bool __gnu_parallel::__cas_omp` (volatile `_Tp *``__ptr`, `_Tp` `__comparand`, `_Tp` `__replacement`)
- `template<typename _Tp >`
`bool __gnu_parallel::__compare_and_swap` (volatile `_Tp *``__ptr`, `_Tp` `__comparand`, `_Tp` `__replacement`)
- `template<typename _Tp >`
`_Tp __gnu_parallel::__fetch_and_add` (volatile `_Tp *``__ptr`, `_Tp` `__addend`)
- `void __gnu_parallel::__yield` ()

7.691.1 Detailed Description

Compatibility layer, mostly concerned with atomic operations.

This file is a GNU parallel extension to the Standard C++ Library and contains implementation details for the library's internal use.

7.692 compatibility.h File Reference

7.692.1 Detailed Description

This is an internal header file, included by other library sources. You should not attempt to use it directly.

7.693 cpu_defines.h File Reference

7.693.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

7.694 ctype_base.h File Reference

Classes

- struct [std::ctype_base](#)

Namespaces

- namespace [std](#)

7.694.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.695 ctype_inline.h File Reference

Namespaces

- namespace [std](#)

7.695.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.696 cxxabi_tweaks.h File Reference

Macros

- `#define _GLIBCXX_CXA_VEC_CTOR_RETURN(x)`
- `#define _GLIBCXX_GUARD_BIT`
- `#define _GLIBCXX_GUARD_PENDING_BIT`
- `#define _GLIBCXX_GUARD_SET(x)`
- `#define _GLIBCXX_GUARD_TEST(x)`
- `#define _GLIBCXX_GUARD_WAITING_BIT`

Typedefs

- `typedef void __cxxabiv1::__cxa_ctor_return_type`
- `typedef void __cxxabiv1::__cxa_vec_ctor_return_type`

Variables

- `__extension__ typedef int __cxxabiv1::__guard`

7.696.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cxxabi.h>`.

7.697 error_constants.h File Reference

Namespaces

- namespace `std`

Enumerations

- enum class `errc` {
 `address_family_not_supported` , `address_in_use` , `address_not_available` , `already_connected` ,
 `argument_list_too_long` , `argument_out_of_domain` , `bad_address` , `bad_file_descriptor` ,
 `broken_pipe` , `connection_aborted` , `connection_already_in_progress` , `connection_refused` ,
 `connection_reset` , `cross_device_link` , `destination_address_required` , `device_or_resource_busy` ,
 `directory_not_empty` , `executable_format_error` , `file_exists` , `file_too_large` ,
 `filename_too_long` , `function_not_supported` , `host_unreachable` , `illegal_byte_sequence` ,
 `inappropriate_io_control_operation` , `interrupted` , `invalid_argument` , `invalid_seek` ,
 `io_error` , `is_a_directory` , `message_size` , `network_down` ,
 `network_reset` , `network_unreachable` , `no_buffer_space` , `no_child_process` ,
 `no_lock_available` , `no_message` , `no_protocol_option` , `no_space_on_device` ,
 `no_such_device_or_address` , `no_such_device` , `no_such_file_or_directory` , `no_such_process` ,
 `not_a_directory` , `not_a_socket` , `not_connected` , `not_enough_memory` ,
 `operation_in_progress` , `operation_not_permitted` , `operation_not_supported` , `operation_would_block` ,
 `permission_denied` , `protocol_not_supported` , `read_only_file_system` , `resource_deadlock_would_occur` ,
 `resource_unavailable_try_again` , `result_out_of_range` , `timed_out` , `too_many_files_open_in_system` ,
 `too_many_files_open` , `too_many_links` , `too_many_symbolic_link_levels` , `wrong_protocol_type` }

7.697.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<system_error>`.

7.698 extc++.h File Reference

7.698.1 Detailed Description

This is an implementation file for a precompiled header.

7.699 messages_members.h File Reference

Namespaces

- namespace [std](#)

7.699.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

7.700 opt_random.h File Reference

Namespaces

- namespace [std](#)

7.700.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

7.701 os_defines.h File Reference

Macros

- `#define __NO_CTYPE`
- `#define _GLIBCXX_NATIVE_THREAD_ID`
- `#define _GLIBCXX_NO_OBSOLETE_ISINF_ISNAN_DYNAMIC`

7.701.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

7.702 stdc++.h File Reference

7.702.1 Detailed Description

This is an implementation file for a precompiled header.

7.703 stdtr1c++.h File Reference

7.703.1 Detailed Description

This is an implementation file for a precompiled header.

7.704 time_members.h File Reference

Namespaces

- namespace [std](#)

7.704.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

Index

`_AlgorithmStrategy`
 [__gnu_parallel](#), 468

`_BALLOC_ALIGN_BYTES`
 [bitmap_allocator.h](#), 3399

`_BinIndex`
 [__gnu_parallel](#), 467

`_Bit_scan_forward`
 [__gnu_cxx](#), 440

`_CASable`
 [__gnu_parallel](#), 467

`_CASable_bits`
 [__gnu_parallel](#), 508

`_CASable_mask`
 [__gnu_parallel](#), 508

`_Construct`
 [std](#), 643

`_DRandomShufflingGlobalData`
 [__gnu_parallel::DRandomShufflingGlobalData<](#)
 [_RAIter >](#), 946

`_Destroy`
 [std](#), 643, 644

`_Destroy_n`
 [std](#), 644

`_Distance_precision`
 [__gnu_debug](#), 457

`_FindAlgorithm`
 [__gnu_parallel](#), 468

`_Find_first`
 [SGI](#), 158

`_Find_next`
 [SGI](#), 158

`_GLIBCXX_BAL_QUICKSORT`
 [features.h](#), 3511

`_GLIBCXX_CALL`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_DEBUG_VERIFY_COND_AT`
 [macros.h](#), 3376

`_GLIBCXX_DEQUE_BUF_SIZE`
 [stl_deque.h](#), 3288

`_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`
 [features.h](#), 3511

`_GLIBCXX_FIND_EQUAL_SPLIT`
 [features.h](#), 3511

`_GLIBCXX_FIND_GROWING_BLOCKS`
 [features.h](#), 3512

`_GLIBCXX_MERGESORT`
 [features.h](#), 3512

`_GLIBCXX_PARALLEL_ASSERTIONS`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_PARALLEL_CONDITION`
 [settings.h](#), 3527

`_GLIBCXX_PARALLEL_LENGTH`
 [multiway_merge.h](#), 3520

`_GLIBCXX_QUICKSORT`
 [features.h](#), 3512

`_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_SCALE_DOWN_FPU`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_TREE_DYNAMIC_BALANCING`
 [features.h](#), 3512

`_GLIBCXX_TREE_FULL_COPY`
 [features.h](#), 3512

`_GLIBCXX_TREE_INITIAL_SPLITTING`
 [features.h](#), 3512

`_GLIBCXX_VERBOSE_LEVEL`
 [compiletime_settings.h](#), 3510

`_GLIBCXX_VOLATILE`
 [partition.h](#), 3524
 [queue.h](#), 3524

`_GuardedIterator`
 [__gnu_parallel::GuardedIterator<](#) [_RAIter](#), [_Compare >](#), 960

`_LoserTreeBase`
 [__gnu_parallel::LoserTreeBase<](#) [_Tp](#), [_Compare >](#), 988

`_M_allocate_and_copy`
 [std::vector<](#) [_Tp](#), [_Alloc >](#), 3108

`_M_allocate_single_object`
 [__gnu_cxx::bitmap_allocator<](#) [_Tp >](#), 1763

`_M_attach`
 [__gnu_debug::Safe_iterator<](#) [_Iterator](#), [_Sequence](#), [_Category >](#), 1033
 [__gnu_debug::Safe_iterator_base](#), 1041
 [__gnu_debug::Safe_local_iterator<](#) [_Iterator](#), [_Sequence >](#), 1046
 [__gnu_debug::Safe_local_iterator_base](#), 1053

`_M_attach_single`
 [__gnu_debug::Safe_iterator<](#) [_Iterator](#), [_Sequence](#), [_Category >](#), 1034
 [__gnu_debug::Safe_iterator_base](#), 1041
 [__gnu_debug::Safe_local_iterator<](#) [_Iterator](#), [_Sequence >](#), 1046, 1047
 [__gnu_debug::Safe_local_iterator_base](#), 1053

`_M_attached_to`
 [__gnu_debug::Safe_iterator<](#) [_Iterator](#), [_Sequence](#), [_Category >](#), 1034
 [__gnu_debug::Safe_iterator_base](#), 1041
 [__gnu_debug::Safe_local_iterator<](#) [_Iterator](#), [_Sequence >](#), 1047

- __gnu_debug::Safe_local_iterator_base, 1053
- _M_before_dereferenceable
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1034
- _M_begin
 - __gnu_parallel::Piece< _DifferenceTp >, 1013
- _M_bin_proc
 - __gnu_parallel::DRandomShufflingGlobalData< _RAIter >, 946
- _M_bins_begin
 - __gnu_parallel::DRSSorterPU< _RAIter, _RandomNumberGenerator >, 948
- _M_buf
 - __gnu_cxx::enc_filebuf< _CharT >, 2037
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2822
 - std::basic_filebuf< _CharT, _Traits >, 1188
- _M_buf_locale
 - __gnu_cxx::enc_filebuf< _CharT >, 2037
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2822
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2839
 - std::basic_filebuf< _CharT, _Traits >, 1188
 - std::basic_streambuf< _CharT, _Traits >, 1555
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1677
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3140
- _M_buf_size
 - __gnu_cxx::enc_filebuf< _CharT >, 2037
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2822
 - std::basic_filebuf< _CharT, _Traits >, 1188
- _M_can_compare
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1034
 - __gnu_debug::Safe_iterator_base, 1041
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1047
 - __gnu_debug::Safe_local_iterator_base, 1054
- _M_clear
 - __gnu_cxx::free_list, 2094
- _M_comp
 - __gnu_parallel::LoserTreeBase< _Tp, _Compare >, 989
- _M_const_iterators
 - __gnu_debug::Safe_forward_list< _SafeSequence >, 1029
 - __gnu_debug::Safe_node_sequence< _Sequence >, 1058
 - __gnu_debug::Safe_sequence< _Sequence >, 1061
 - __gnu_debug::Safe_sequence_base, 1063
 - __gnu_debug::Safe_unordered_container< _Container >, 1066
 - __gnu_debug::Safe_unordered_container_base, 1069
- _M_const_local_iterators
 - __gnu_debug::Safe_unordered_container< _Container >, 1066
 - __gnu_debug::Safe_unordered_container_base, 1069
- _M_create_node
 - std::list< _Tp, _Alloc >, 2291
- _M_create_pback
 - __gnu_cxx::enc_filebuf< _CharT >, 2023
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2808
 - std::basic_filebuf< _CharT, _Traits >, 1174
- _M_deallocate_single_object
 - __gnu_cxx::bitmap_allocator< _Tp >, 1763
- _M_dereferenceable
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1034
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1047
- _M_destroy_pback
 - __gnu_cxx::enc_filebuf< _CharT >, 2023
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2808
 - std::basic_filebuf< _CharT, _Traits >, 1174
- _M_detach
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1034
 - __gnu_debug::Safe_iterator_base, 1041
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1047
 - __gnu_debug::Safe_local_iterator_base, 1054
- _M_detach_all
 - __gnu_debug::Safe_forward_list< _SafeSequence >, 1028
 - __gnu_debug::Safe_node_sequence< _Sequence >, 1057
 - __gnu_debug::Safe_sequence< _Sequence >, 1060
 - __gnu_debug::Safe_sequence_base, 1062
 - __gnu_debug::Safe_unordered_container< _Container >, 1065
 - __gnu_debug::Safe_unordered_container_base, 1068
- _M_detach_single
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1035
 - __gnu_debug::Safe_iterator_base, 1041
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1047
 - __gnu_debug::Safe_local_iterator_base, 1054
- _M_detach_singular
 - __gnu_debug::Safe_forward_list< _SafeSequence >, 1028
 - __gnu_debug::Safe_node_sequence< _Sequence >, 1057
 - __gnu_debug::Safe_sequence< _Sequence >, 1060

- __gnu_debug::_Safe_sequence_base, [1062](#)
 - __gnu_debug::_Safe_unordered_container< _Container >, [1065](#)
 - __gnu_debug::_Safe_unordered_container_base, [1068](#)
- _M_dist
 - __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >, [946](#)
- _M_elements_leftover
 - __gnu_parallel::_QSBThreadLocal< _RAIter >, [1021](#)
- _M_end
 - __gnu_parallel::_Piece< _DifferenceTp >, [1013](#)
- _M_ext_buf
 - __gnu_cxx::enc_filebuf< _CharT >, [2037](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2822](#)
 - std::basic_filebuf< _CharT, _Traits >, [1188](#)
- _M_ext_buf_size
 - __gnu_cxx::enc_filebuf< _CharT >, [2037](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2823](#)
 - std::basic_filebuf< _CharT, _Traits >, [1188](#)
- _M_ext_next
 - __gnu_cxx::enc_filebuf< _CharT >, [2038](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2823](#)
 - std::basic_filebuf< _CharT, _Traits >, [1188](#)
- _M_fill_initialize
 - std::deque< _Tp, _Alloc >, [1971](#)
- _M_finish_iterator
 - __gnu_parallel::__accumulate_selector< _It >, [802](#)
 - __gnu_parallel::__adjacent_difference_selector< _It >, [802](#)
 - __gnu_parallel::__count_if_selector< _It, _Diff >, [823](#)
 - __gnu_parallel::__count_selector< _It, _Diff >, [824](#)
 - __gnu_parallel::__fill_selector< _It >, [841](#)
 - __gnu_parallel::__for_each_selector< _It >, [845](#)
 - __gnu_parallel::__generate_selector< _It >, [847](#)
 - __gnu_parallel::__generic_for_each_selector< _It >, [849](#)
 - __gnu_parallel::__identity_selector< _It >, [850](#)
 - __gnu_parallel::__inner_product_selector< _It, _It2, _Tp >, [852](#)
 - __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >, [875](#)
 - __gnu_parallel::__replace_selector< _It, _Tp >, [877](#)
 - __gnu_parallel::__transform1_selector< _It >, [879](#)
 - __gnu_parallel::__transform2_selector< _It >, [880](#)
- _M_first
 - __gnu_parallel::_Job< _DifferenceTp >, [970](#)
- _M_first_insert
 - __gnu_parallel::_LoserTreeBase< _Tp, _Compare >, [989](#)
- _M_gcount
 - std::basic_fstream< _CharT, _Traits >, [1235](#)
- std::basic_ifstream< _CharT, _Traits >, [1278](#)
- std::basic_iostream< _CharT, _Traits >, [1348](#)
- std::basic_istream< _CharT, _Traits >, [1385](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1426](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1721](#)
- _M_get
 - __gnu_cxx::free_list, [2094](#)
- _M_get_mutex
 - __gnu_debug::_Safe_forward_list< _SafeSequence >, [1028](#)
 - __gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >, [1035](#)
 - __gnu_debug::_Safe_iterator_base, [1041](#)
 - __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >, [1047](#)
 - __gnu_debug::_Safe_local_iterator_base, [1054](#)
 - __gnu_debug::_Safe_node_sequence< _Sequence >, [1057](#)
 - __gnu_debug::_Safe_sequence< _Sequence >, [1060](#)
 - __gnu_debug::_Safe_sequence_base, [1062](#)
 - __gnu_debug::_Safe_unordered_container< _Container >, [1065](#)
 - __gnu_debug::_Safe_unordered_container_base, [1068](#)
- _M_get_result
 - std::__basic_future< _Res >, [815](#)
 - std::future< _Res >, [2104](#)
 - std::future< _Res & >, [2106](#)
 - std::future< void >, [2107](#)
 - std::shared_future< _Res >, [2767](#)
 - std::shared_future< _Res & >, [2769](#)
 - std::shared_future< void >, [2771](#)
- _M_getloc
 - std::basic_fstream< _CharT, _Traits >, [1200](#)
 - std::basic_ifstream< _CharT, _Traits >, [1251](#)
 - std::basic_ios< _CharT, _Traits >, [1291](#)
 - std::basic_iostream< _CharT, _Traits >, [1315](#)
 - std::basic_istream< _CharT, _Traits >, [1360](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1398](#)
 - std::basic_ofstream< _CharT, _Traits >, [1440](#)
 - std::basic_ostream< _CharT, _Traits >, [1472](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1505](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1687](#)
 - std::ios_base, [2192](#)
- _M_global
 - __gnu_parallel::_QSBThreadLocal< _RAIter >, [1021](#)
- _M_in_beg

- `__gnu_cxx::enc_filebuf< _CharT >`, 2038
- `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
- `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
- `std::basic_filebuf< _CharT, _Traits >`, 1188
- `std::basic_streambuf< _CharT, _Traits >`, 1555
- `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
- `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3140
- `_M_in_cur`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
 - `std::basic_filebuf< _CharT, _Traits >`, 1188
 - `std::basic_streambuf< _CharT, _Traits >`, 1555
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3141
- `_M_in_end`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
 - `std::basic_streambuf< _CharT, _Traits >`, 1555
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3141
- `_M_in_same_bucket`
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
- `_M_incrementable`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1035
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
- `_M_initial`
 - `__gnu_parallel::QSBThreadLocal< _RAIter >`, 1021
- `_M_initialize_map`
 - `std::Deque_base< _Tp, _Alloc >`, 943
- `_M_insert`
 - `__gnu_cxx::free_list`, 2094
- `_M_invalidate`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1035
 - `__gnu_debug::Safe_iterator_base`, 1042
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
 - `__gnu_debug::Safe_local_iterator_base`, 1054
- `_M_invalidate_all`
 - `__gnu_debug::Safe_forward_list< _SafeSequence >`, 1028
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1057
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1060
- `__gnu_debug::Safe_sequence_base`, 1063
- `__gnu_debug::Safe_unordered_container< _Container >`, 1065
- `__gnu_debug::Safe_unordered_container_base`, 1068
- `_M_invalidate_if`
 - `__gnu_debug::Safe_forward_list< _SafeSequence >`, 1028
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1057
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1060
 - `__gnu_debug::Safe_unordered_container< _Container >`, 1065
- `_M_invalidate_local_if`
 - `__gnu_debug::Safe_unordered_container< _Container >`, 1066
- `_M_is_before_begin`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1035
- `_M_is_begin`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1035
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
- `_M_is_beginnest`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1035
- `_M_is_end`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1036
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
- `_M_iterators`
 - `__gnu_debug::Safe_forward_list< _SafeSequence >`, 1029
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1058
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1061
 - `__gnu_debug::Safe_sequence_base`, 1063
 - `__gnu_debug::Safe_unordered_container< _Container >`, 1066
 - `__gnu_debug::Safe_unordered_container_base`, 1069
- `_M_key`
 - `__gnu_parallel::LoserTreeBase< _Tp, _Compare >::Loser`, 982
- `_M_last`
 - `__gnu_parallel::Job< _DifferenceTp >`, 970
- `_M_leftover_parts`
 - `__gnu_parallel::QSBThreadLocal< _RAIter >`, 1021

- `_M_load`
 - `__gnu_parallel::_Job< _DifferenceTp >`, 971
- `_M_local_iterators`
 - `__gnu_debug::_Safe_unordered_container< _Container >`, 1066
 - `__gnu_debug::_Safe_unordered_container_base`, 1069
- `_M_log_k`
 - `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >`, 984
 - `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >`, 989
- `_M_losers`
 - `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >`, 989
- `_M_merge_equal`
 - `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`, 2675
- `_M_merge_unique`
 - `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`, 2675
- `_M_mode`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
- `_M_new_elements_at_back`
 - `std::deque< _Tp, _Alloc >`, 1971
- `_M_new_elements_at_front`
 - `std::deque< _Tp, _Alloc >`, 1971
- `_M_next`
 - `__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >`, 1038
 - `__gnu_debug::_Safe_iterator_base`, 1042
 - `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >`, 1051
 - `__gnu_debug::_Safe_local_iterator_base`, 1055
- `_M_num_bins`
 - `__gnu_parallel::_DRandomShufflingGlobalData< _RAIter >`, 947
- `_M_num_bits`
 - `__gnu_parallel::_DRandomShufflingGlobalData< _RAIter >`, 947
- `_M_num_threads`
 - `__gnu_parallel::_DRSSorterPU< _RAIter, _RandomNumberGenerator >`, 948
 - `__gnu_parallel::_PMWMSSortingData< _RAIter >`, 1015
 - `__gnu_parallel::_QSBThreadLocal< _RAIter >`, 1021
- `_M_offsets`
 - `__gnu_parallel::_PMWMSSortingData< _RAIter >`, 1015
- `_M_out_beg`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
 - `std::basic_streambuf< _CharT, _Traits >`, 1555
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3141
- `_M_out_cur`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
 - `std::basic_streambuf< _CharT, _Traits >`, 1555
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1677
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3141
- `_M_out_end`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2823
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2839
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
 - `std::basic_streambuf< _CharT, _Traits >`, 1555
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1678
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3141
- `_M_pback`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2038
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2824
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
- `_M_pback_cur_save`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2039
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2824
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
- `_M_pback_end_save`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2039
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2824
 - `std::basic_filebuf< _CharT, _Traits >`, 1189
- `_M_pback_init`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2039
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2824
 - `std::basic_filebuf< _CharT, _Traits >`, 1190
- `_M_pieces`
 - `__gnu_parallel::_PMWMSSortingData< _RAIter >`, 1015
- `_M_pop_back_aux`
 - `std::deque< _Tp, _Alloc >`, 1972
- `_M_pop_front_aux`
 - `std::deque< _Tp, _Alloc >`, 1972
- `_M_prior`
 - `__gnu_debug::_Safe_iterator< _Iterator, _Sequence, _Category >`, 1038
 - `__gnu_debug::_Safe_iterator_base`, 1042

- __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1051
- __gnu_debug::Safe_local_iterator_base, 1055
- _M_push_back_aux
 - std::deque< _Tp, _Alloc >, 1972
- _M_push_front_aux
 - std::deque< _Tp, _Alloc >, 1972
- _M_range_check
 - std::deque< _Tp, _Alloc >, 1972
 - std::vector< _Tp, _Alloc >, 3108
- _M_range_initialize
 - std::deque< _Tp, _Alloc >, 1972, 1973
- _M_reading
 - __gnu_cxx::enc_filebuf< _CharT >, 2039
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2824
 - std::basic_filebuf< _CharT, _Traits >, 1190
- _M_reallocate_map
 - std::deque< _Tp, _Alloc >, 1973
- _M_reinsert_node_equal
 - __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >, 2675
- _M_reinsert_node_hint_equal
 - __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >, 2675
- _M_reinsert_node_hint_unique
 - __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >, 2675
- _M_reinsert_node_unique
 - __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >, 2676
- _M_reserve_elements_at_back
 - std::deque< _Tp, _Alloc >, 1973
- _M_reserve_elements_at_front
 - std::deque< _Tp, _Alloc >, 1974
- _M_reserve_map_at_back
 - std::deque< _Tp, _Alloc >, 1974
- _M_reserve_map_at_front
 - std::deque< _Tp, _Alloc >, 1974
- _M_reset
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1036
 - __gnu_debug::Safe_iterator_base, 1042
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1048
 - __gnu_debug::Safe_local_iterator_base, 1054
- _M_revalidate_singular
 - __gnu_debug::Safe_forward_list< _SafeSequence >, 1028
 - __gnu_debug::Safe_node_sequence< _Sequence >, 1057
 - __gnu_debug::Safe_sequence< _Sequence >, 1060
 - __gnu_debug::Safe_sequence_base, 1063
- __gnu_debug::Safe_unordered_container< _Container >, 1066
- __gnu_debug::Safe_unordered_container_base, 1069
- _M_samples
 - __gnu_parallel::PMWMSortingData< _RAIter >, 1015
- _M_sd
 - __gnu_parallel::DRSSorterPU< _RAIter, _RandomNumberGenerator >, 948
- _M_seed
 - __gnu_parallel::DRSSorterPU< _RAIter, _RandomNumberGenerator >, 948
- _M_sequence
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1038
 - __gnu_debug::Safe_iterator_base, 1042
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1051
 - __gnu_debug::Safe_local_iterator_base, 1055
- _M_sequential_algorithm
 - __gnu_parallel::adjacent_find_selector, 803
 - __gnu_parallel::find_first_of_selector< _FIterator >, 842
 - __gnu_parallel::find_if_selector, 843
 - __gnu_parallel::mismatch_selector, 856
- _M_set_buffer
 - __gnu_cxx::enc_filebuf< _CharT >, 2023
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2808
 - std::basic_filebuf< _CharT, _Traits >, 1174
- _M_set_node
 - std::Deque_iterator< _Tp, _Ref, _Ptr >, 945
- _M_singular
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1036
 - __gnu_debug::Safe_iterator_base, 1042
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1048
 - __gnu_debug::Safe_local_iterator_base, 1054
- _M_source
 - __gnu_parallel::DRandomShufflingGlobalData< _RAIter >, 947
 - __gnu_parallel::LoserTreeBase< _Tp, _Compare >::Loser, 982
 - __gnu_parallel::PMWMSortingData< _RAIter >, 1015
- _M_starts
 - __gnu_parallel::DRandomShufflingGlobalData< _RAIter >, 947
 - __gnu_parallel::PMWMSortingData< _RAIter >, 1015
- _M_sup
 - __gnu_parallel::LoserTreeBase< _Tp, _Compare >::Loser, 982

- `_M_swap`
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1057
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1060
 - `__gnu_debug::Safe_sequence_base`, 1063
 - `__gnu_debug::Safe_unordered_container< _Container >`, 1066
 - `__gnu_debug::Safe_unordered_container_base`, 1069
- `_M_temporaries`
 - `__gnu_parallel::DRandomShufflingGlobalData< _RAIter >`, 947
- `_M_temporary`
 - `__gnu_parallel::PMWMSSortingData< _RAIter >`, 1015
- `_M_transfer_from_if`
 - `__gnu_debug::Safe_forward_list< _SafeSequence >`, 1029
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1057
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1060
- `_M_unlink`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1036
 - `__gnu_debug::Safe_iterator_base`, 1042
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1048
 - `__gnu_debug::Safe_local_iterator_base`, 1054
- `_M_use_pointer`
 - `__gnu_parallel::LoserTreeTraits< _Tp >`, 995
- `_M_version`
 - `__gnu_debug::Safe_forward_list< _SafeSequence >`, 1029
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1039
 - `__gnu_debug::Safe_iterator_base`, 1043
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1051
 - `__gnu_debug::Safe_local_iterator_base`, 1055
 - `__gnu_debug::Safe_node_sequence< _Sequence >`, 1058
 - `__gnu_debug::Safe_sequence< _Sequence >`, 1061
 - `__gnu_debug::Safe_sequence_base`, 1063
 - `__gnu_debug::Safe_unordered_container< _Container >`, 1067
 - `__gnu_debug::Safe_unordered_container_base`, 1069
- `_M_w`
 - `std::Base_bitset< _Nw >`, 934
 - `std::tr2::__dynamic_bitset_base< _WordT, _Alloc >`, 839
- `_MultiwayMergeAlgorithm`
 - `__gnu_parallel`, 468
- `_Opcode`
 - Base and Implementation Classes, 301
- `_Parallelism`
 - `__gnu_parallel`, 468
- `_PartialSumAlgorithm`
 - `__gnu_parallel`, 468
- `_Piece`
 - `__gnu_parallel::QSBThreadLocal< _RAIter >`, 1020
- `_PseudoSequence`
 - `__gnu_parallel::PseudoSequence< _Tp, _DifferenceTp >`, 1019
- `_QSBThreadLocal`
 - `__gnu_parallel::QSBThreadLocal< _RAIter >`, 1021
- `_RandomNumber`
 - `__gnu_parallel::RandomNumber`, 1022
- `_RestrictedBoundedConcurrentQueue`
 - `__gnu_parallel::RestrictedBoundedConcurrentQueue< _Tp >`, 1025
- `_S_constant`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1036
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1049
- `_Safe_iterator`
 - `__gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >`, 1032, 1033
- `_Safe_iterator_base`
 - `__gnu_debug::Safe_iterator_base`, 1040
- `_Safe_local_iterator`
 - `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >`, 1045, 1046
- `_Safe_local_iterator_base`
 - `__gnu_debug::Safe_local_iterator_base`, 1053
- `_SequenceIndex`
 - `__gnu_parallel`, 468
- `_SortAlgorithm`
 - `__gnu_parallel`, 468
- `_SplittingAlgorithm`
 - `__gnu_parallel`, 469
- `_Temporary_buffer`
 - `std::Temporary_buffer< _ForwardIterator, _Tp >`, 1083
- `_ThreadIndex`
 - `__gnu_parallel`, 468
- `_TokenT`
 - `std::__detail::Scanner< _CharT >`, 1072
- `_Unchecked_flip`
 - SGI, 159
- `_Unchecked_reset`
 - SGI, 159

- `__Unchecked_set`
 - SGL, [159](#)
- `__Unchecked_test`
 - SGL, [159](#)
- `__addressof`
 - Utilities, [341](#)
- `__allocator_base`
 - Allocators, [368](#)
- `__base`
 - `__gnu_debug`, [457](#)
- `__begin1_iterator`
 - `__gnu_parallel::__inner_product_selector< _It, _It2, _Tp >`, [852](#)
- `__begin2_iterator`
 - `__gnu_parallel::__inner_product_selector< _It, _It2, _Tp >`, [852](#)
- `__bins_end`
 - `__gnu_parallel::__DRSSorterPU< _RAIter, _RandomNumberGenerator >`, [948](#)
- `__bit_allocate`
 - `__gnu_cxx::__detail`, [450](#)
- `__bit_free`
 - `__gnu_cxx::__detail`, [450](#)
- `__calc_borders`
 - `__gnu_parallel`, [469](#)
- `__check_singular`
 - `__gnu_debug`, [457](#)
- `__check_singular_aux`
 - `__gnu_debug`, [457](#)
- `__check_string`
 - `__gnu_debug`, [457](#), [458](#)
- `__compare_and_swap`
 - `__gnu_parallel`, [469](#)
- `__cpp_lib_experimental_detect`
 - Detection idiom, [313](#)
- `__cpp_lib_remove_cvref`
 - Metaprogramming, [396](#)
- `__cpp_lib_tuples_by_type`
 - `stl_pair.h`, [3304](#)
- `__cpp_lib_type_identity`
 - Metaprogramming, [396](#)
- `__ctype_type`
 - `std::basic_ios< _CharT, _Traits >`, [1287](#)
- `__cxa_demangle`
 - `cxxabi.h`, [3151](#)
- `__cxxabiv1::__forced_unwind`, [845](#)
- `__decode2`
 - `__gnu_parallel`, [469](#)
- `__delete_min_insert`
 - `__gnu_parallel::__LoserTree< __stable, _Tp, _Compare >`, [983](#)
 - `__gnu_parallel::__LoserTree< false, _Tp, _Compare >`, [985](#)
- `__determine_samples`
 - `__gnu_parallel`, [470](#)
- `__encode2`
 - `__gnu_parallel`, [470](#)
- `__equally_split`
 - `__gnu_parallel`, [471](#)
- `__equally_split_point`
 - `__gnu_parallel`, [471](#)
- `__fetch_and_add`
 - `__gnu_parallel`, [471](#)
- `__final_insertion_sort`
 - `std`, [635](#)
- `__find_if`
 - `std`, [635](#)
- `__find_if_not`
 - `std`, [635](#)
- `__find_if_not_n`
 - `std`, [636](#)
- `__find_template`
 - `__gnu_parallel`, [472–474](#)
- `__for_each_template_random_access`
 - `__gnu_parallel`, [474](#)
- `__for_each_template_random_access_ed`
 - `__gnu_parallel`, [475](#)
- `__for_each_template_random_access_omp_loop`
 - `__gnu_parallel`, [476](#)
- `__for_each_template_random_access_omp_loop_static`
 - `__gnu_parallel`, [476](#)
- `__for_each_template_random_access_workstealing`
 - `__gnu_parallel`, [477](#)
- `__foreign_iterator_aux2`
 - `__gnu_debug`, [458](#)
- `__from_chars_alnum`
 - `std::__detail`, [723](#)
- `__from_chars_pow2_base`
 - `std::__detail`, [723](#)
- `__gcd`
 - `std`, [636](#)
- `__gen_two_uniform_ints`
 - `std`, [636](#)
- `__genrand_bits`
 - `__gnu_parallel::__RandomNumber`, [1023](#)
- `__get_distance`
 - `__gnu_debug`, [458](#)
- `__get_min_source`
 - `__gnu_parallel::__LoserTree< __stable, _Tp, _Compare >`, [983](#)
 - `__gnu_parallel::__LoserTree< false, _Tp, _Compare >`, [986](#)
 - `__gnu_parallel::__LoserTreeBase< _Tp, _Compare >`, [988](#)
- `__get_num_threads`
 - `__gnu_parallel::balanced_quicksort_tag`, [1167](#)
 - `__gnu_parallel::balanced_tag`, [1168](#)
 - `__gnu_parallel::default_parallel_tag`, [1958](#)

- __gnu_parallel::exact_tag, 2051
- __gnu_parallel::multiway_mergesort_exact_tag, 2481
- __gnu_parallel::multiway_mergesort_sampling_tag, 2482
- __gnu_parallel::multiway_mergesort_tag, 2483
- __gnu_parallel::omp_loop_static_tag, 2562
- __gnu_parallel::omp_loop_tag, 2563
- __gnu_parallel::parallel_tag, 2598
- __gnu_parallel::quicksort_tag, 2654
- __gnu_parallel::sampling_tag, 2732
- __gnu_parallel::unbalanced_tag, 2956
- __glibcxx_check_erase
 - macros.h, 3374
- __glibcxx_check_erase_after
 - macros.h, 3374
- __glibcxx_check_erase_range
 - macros.h, 3374
- __glibcxx_check_erase_range_after
 - macros.h, 3374
- __glibcxx_check_heap_pred
 - macros.h, 3374
- __glibcxx_check_insert
 - macros.h, 3374
- __glibcxx_check_insert_after
 - macros.h, 3374
- __glibcxx_check_insert_range
 - macros.h, 3375
- __glibcxx_check_insert_range_after
 - macros.h, 3375
- __glibcxx_check_partitioned_lower
 - macros.h, 3375
- __glibcxx_check_partitioned_lower_pred
 - macros.h, 3375
- __glibcxx_check_partitioned_upper_pred
 - macros.h, 3375
- __glibcxx_check_sorted_pred
 - macros.h, 3376
- __gnu_cxx, 424
 - _Bit_scan_forward, 440
 - __int_traits, 439
 - __static_pointer_cast, 439
 - operator!=, 440, 441
 - operator<, 443, 444
 - operator<=, 444, 445
 - operator>, 446, 447
 - operator>=, 448
 - operator+, 441–443
 - operator==, 445, 446
 - swap, 449
- __gnu_cxx::Caster<_ToType>, 938
- __gnu_cxx::Char_types<_CharT>, 938
- __gnu_cxx::ExtPtr_allocator<_Tp>, 952
- __gnu_cxx::Invalid_type, 964
- __gnu_cxx::Pointer_adapter<_Storage_policy>, 1016
- __gnu_cxx::Relative_pointer_impl<_Tp>, 1023
- __gnu_cxx::Relative_pointer_impl<const _Tp>, 1024
- __gnu_cxx::Std_pointer_impl<_Tp>, 1081
- __gnu_cxx::Unqualified_type<_Tp>, 1085
- __gnu_cxx::__alloc_traits<_Alloc, typename>, 804
 - allocate, 806, 807
 - const_void_pointer, 806
 - construct, 808
 - deallocate, 808, 809
 - destroy, 809
 - is_always_equal, 806
 - max_size, 809
 - propagate_on_container_copy_assignment, 806
 - propagate_on_container_move_assignment, 806
 - propagate_on_container_swap, 806
 - select_on_container_copy_construction, 810
 - void_pointer, 806
- __gnu_cxx::__common_pool_policy<_PoolTp, _Thread>, 821
- __gnu_cxx::__detail, 449
 - _bit_allocate, 450
 - _bit_free, 450
 - _num_bitmaps, 450
 - _num_blocks, 450
- __gnu_cxx::__detail::Bitmap_counter<_Tp>, 937
- __gnu_cxx::__detail::Ffit_finder<_Tp>, 953
- __gnu_cxx::__detail::__mini_vector<_Tp>, 854
- __gnu_cxx::__mt_alloc<_Tp, _Poolp>, 856
- __gnu_cxx::__mt_alloc_base<_Tp>, 858
- __gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread>, 866
- __gnu_cxx::__pool<_Thread>, 866
- __gnu_cxx::__pool<false>, 866
- __gnu_cxx::__pool<true>, 868
- __gnu_cxx::__pool_alloc<_Tp>, 869
- __gnu_cxx::__pool_alloc_base, 870
- __gnu_cxx::__pool_base, 871
- __gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>, 871
- __gnu_cxx::__scoped_lock, 877
- __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>, 881
 - _versa_string, 884–887
 - ~__versa_string, 887
- append, 888–890
- assign, 890–893
- at, 894
- back, 895
- begin, 895
- c_str, 895
- capacity, 895
- cbegin, 896
- cend, 896

- clear, [896](#)
- compare, [896–899](#)
- copy, [899](#)
- crbegin, [900](#)
- crend, [900](#)
- data, [900](#)
- empty, [900](#)
- end, [900](#), [901](#)
- erase, [901](#), [902](#)
- find, [902](#), [903](#)
- find_first_not_of, [904](#), [905](#)
- find_first_of, [905–907](#)
- find_last_not_of, [907–909](#)
- find_last_of, [909](#), [910](#)
- front, [911](#)
- get_allocator, [911](#)
- insert, [911–916](#)
- length, [916](#)
- max_size, [916](#)
- npos, [933](#)
- operator+=, [917](#), [918](#)
- operator=, [918](#), [919](#)
- operator[], [919](#), [920](#)
- pop_back, [920](#)
- push_back, [920](#)
- rbegin, [921](#)
- rend, [921](#)
- replace, [921–927](#)
- reserve, [928](#)
- resize, [929](#)
- rfind, [929–931](#)
- shrink_to_fit, [931](#)
- size, [931](#)
- substr, [932](#)
- swap, [932](#)
- __gnu_cxx::annotate_base, [1111](#)
- __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >, [1739](#)
- argument_type, [1740](#)
- result_type, [1740](#)
- __gnu_cxx::bitmap_allocator< _Tp >, [1762](#)
- _M_allocate_single_object, [1763](#)
- _M_deallocate_single_object, [1763](#)
- __gnu_cxx::char_traits< _CharT >, [1794](#)
- __gnu_cxx::character< _Value, _Int, _St >, [1798](#)
- __gnu_cxx::condition_base, [1857](#)
- __gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 >, [1865](#)
- __gnu_cxx::constant_unary_fun< _Result, _Argument >, [1866](#)
- __gnu_cxx::constant_void_fun< _Result >, [1866](#)
- __gnu_cxx::debug_allocator< _Alloc >, [1948](#)
- __gnu_cxx::enc_filebuf< _CharT >, [2020](#)
- _M_buf, [2037](#)
- _M_buf_locale, [2037](#)
- _M_buf_size, [2037](#)
- _M_create_pback, [2023](#)
- _M_destroy_pback, [2023](#)
- _M_ext_buf, [2037](#)
- _M_ext_buf_size, [2037](#)
- _M_ext_next, [2038](#)
- _M_in_beg, [2038](#)
- _M_in_cur, [2038](#)
- _M_in_end, [2038](#)
- _M_mode, [2038](#)
- _M_out_beg, [2038](#)
- _M_out_cur, [2038](#)
- _M_out_end, [2038](#)
- _M_pback, [2038](#)
- _M_pback_cur_save, [2039](#)
- _M_pback_end_save, [2039](#)
- _M_pback_init, [2039](#)
- _M_reading, [2039](#)
- _M_set_buffer, [2023](#)
- close, [2024](#)
- eback, [2024](#)
- egptr, [2024](#)
- epptr, [2024](#)
- gbump, [2025](#)
- getloc, [2025](#)
- gptr, [2025](#)
- imbue, [2025](#)
- in_avail, [2026](#)
- is_open, [2026](#)
- open, [2026](#), [2027](#)
- overflow, [2028](#)
- pbackfail, [2028](#)
- pbase, [2028](#)
- pbump, [2029](#)
- pptr, [2029](#)
- pubimbue, [2029](#)
- pubseekoff, [2030](#)
- pubseekpos, [2030](#)
- pubsetbuf, [2030](#)
- pubsync, [2030](#)
- sbumpc, [2030](#)
- seekoff, [2031](#)
- seekpos, [2031](#)
- setbuf, [2031](#)
- setg, [2032](#)
- setp, [2032](#)
- sgetc, [2033](#)
- sgetn, [2033](#)
- showmanyc, [2033](#)
- snextc, [2034](#)
- sputbackc, [2034](#)
- sputc, [2034](#)
- sputn, [2035](#)

- sungetc, 2035
- sync, 2035
- uflow, 2035
- underflow, 2036
- xsgetrn, 2036
- xsputn, 2037
- __gnu_cxx::encoding_char_traits< _CharT >, 2039
- __gnu_cxx::encoding_state, 2041
- __gnu_cxx::forced_error, 2070
 - what, 2070
- __gnu_cxx::free_list, 2093
 - _M_clear, 2094
 - _M_get, 2094
 - _M_insert, 2094
- __gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >, 2158
- __gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >, 2160
- __gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc >, 2161
- __gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc >, 2164
- __gnu_cxx::limit_condition, 2276
- __gnu_cxx::limit_condition::always_adjustor, 1110
- __gnu_cxx::limit_condition::limit_adjustor, 2275
- __gnu_cxx::limit_condition::never_adjustor, 2489
- __gnu_cxx::malloc_allocator< _Tp >, 2332
- __gnu_cxx::new_allocator< _Tp >, 2490
- __gnu_cxx::project1st< _Arg1, _Arg2 >, 2645
 - first_argument_type, 2645
 - result_type, 2645
 - second_argument_type, 2645
- __gnu_cxx::project2nd< _Arg1, _Arg2 >, 2645
 - first_argument_type, 2646
 - result_type, 2646
 - second_argument_type, 2646
- __gnu_cxx::random_condition, 2655
- __gnu_cxx::random_condition::always_adjustor, 1110
- __gnu_cxx::random_condition::group_adjustor, 2129
- __gnu_cxx::random_condition::never_adjustor, 2489
- __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >, 2671
 - _M_merge_equal, 2675
 - _M_merge_unique, 2675
 - _M_reinsert_node_equal, 2675
 - _M_reinsert_node_hint_equal, 2675
 - _M_reinsert_node_hint_unique, 2675
 - _M_reinsert_node_unique, 2676
- extract, 2676
- __gnu_cxx::recursive_init_error, 2686
 - what, 2686
- __gnu_cxx::rope< _CharT, _Alloc >, 2712
- __gnu_cxx::select1st< _Pair >, 2735
 - argument_type, 2736
 - result_type, 2736
- __gnu_cxx::select2nd< _Pair >, 2736
 - argument_type, 2736
 - result_type, 2736
- __gnu_cxx::slist< _Tp, _Alloc >, 2793
- __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2804
 - _M_buf, 2822
 - _M_buf_locale, 2822
 - _M_buf_size, 2822
 - _M_create_pback, 2808
 - _M_destroy_pback, 2808
 - _M_ext_buf, 2822
 - _M_ext_buf_size, 2823
 - _M_ext_next, 2823
 - _M_in_beg, 2823
 - _M_in_cur, 2823
 - _M_in_end, 2823
 - _M_mode, 2823
 - _M_out_beg, 2823
 - _M_out_cur, 2823
 - _M_out_end, 2823
 - _M_pback, 2824
 - _M_pback_cur_save, 2824
 - _M_pback_end_save, 2824
 - _M_pback_init, 2824
 - _M_reading, 2824
 - _M_set_buffer, 2808
 - ~stdio_filebuf, 2807
- close, 2808
- eback, 2808
- egptr, 2809
- epptr, 2809
- fd, 2809
- file, 2809
- gbump, 2809
- getloc, 2810
- gptr, 2810
- imbue, 2810
- in_avail, 2811
- is_open, 2811
- open, 2811, 2812
- overflow, 2812
- pbackfail, 2813
- pbase, 2813
- pbump, 2814
- pptr, 2814
- pubimbue, 2814
- pubseekoff, 2815
- pubseekpos, 2815
- pubsetbuf, 2815
- pubsync, 2815
- sbumpc, 2815
- seekoff, 2816
- seekpos, 2816

- setbuf, 2816
- setg, 2817
- setp, 2817
- sgetc, 2818
- sgetn, 2818
- showmanyc, 2818
- snextc, 2819
- sputbackc, 2819
- sputc, 2819
- sputn, 2820
- stdio_filebuf, 2807
- sungetc, 2820
- sync, 2820
- uflow, 2820
- underflow, 2821
- xsggetn, 2821
- xspn, 2822
- __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2825
 - _M_buf_locale, 2839
 - _M_in_beg, 2839
 - _M_in_cur, 2839
 - _M_in_end, 2839
 - _M_out_beg, 2839
 - _M_out_cur, 2839
 - _M_out_end, 2839
 - eback, 2827
 - egptr, 2827
 - eptr, 2827
 - file, 2828
 - gbump, 2828
 - getloc, 2828
 - gptr, 2828
 - imbue, 2829
 - in_avail, 2829
 - overflow, 2829
 - pbackfail, 2830
 - pbase, 2830
 - pbump, 2830
 - pptr, 2831
 - pubimbue, 2831
 - pubseekoff, 2831
 - pubseekpos, 2832
 - pubsetbuf, 2832
 - pubsync, 2832
 - sbumpc, 2832
 - seekoff, 2832
 - seekpos, 2833
 - setbuf, 2833
 - setg, 2833
 - setp, 2834
 - sgetc, 2834
 - sgetn, 2834
 - showmanyc, 2835
 - snextc, 2835
 - sputbackc, 2835
 - sputc, 2836
 - sputn, 2836
 - sungetc, 2836
 - sync, 2837
 - uflow, 2837
 - underflow, 2837
 - xsggetn, 2838
 - xspn, 2838
 - __gnu_cxx::subtractive_rng, 2863
 - argument_type, 2863
 - operator(), 2864
 - result_type, 2863
 - subtractive_rng, 2863
 - __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, 2867
 - ~temporary_buffer, 2868
 - begin, 2869
 - end, 2869
 - requested_size, 2869
 - size, 2869
 - temporary_buffer, 2868
 - __gnu_cxx::throw_allocator_base< _Tp, _Cond >, 2873
 - __gnu_cxx::throw_allocator_limit< _Tp >, 2874
 - __gnu_cxx::throw_allocator_random< _Tp >, 2876
 - __gnu_cxx::throw_value_base< _Cond >, 2878
 - __gnu_cxx::throw_value_limit, 2878
 - __gnu_cxx::throw_value_random, 2880
 - __gnu_cxx::typelist, 450
 - apply_generator, 451
 - __gnu_cxx::unary_compose< _Operation1, _Operation2 >, 2953
 - argument_type, 2953
 - result_type, 2953
 - __gnu_debug, 451
 - _Distance_precision, 457
 - _base, 457
 - _check_singular, 457
 - _check_singular_aux, 457
 - _check_string, 457, 458
 - _foreign_iterator_aux2, 458
 - _get_distance, 458
 - _valid_range, 458, 459
 - _valid_range_aux, 459
 - u16string, 457
 - u32string, 457
 - __gnu_debug:: _After_nth_from< _Iterator >, 933
 - __gnu_debug:: _BeforeBeginHelper< _Sequence >, 936
 - __gnu_debug:: _Equal_to< _Type >, 949
 - __gnu_debug:: _Not_equal_to< _Type >, 1009
 - __gnu_debug:: _Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >, 1025
 - __gnu_debug:: _Safe_forward_list< _SafeSequence >, 1027

- [_M_const_iterators](#), 1029
- [_M_detach_all](#), 1028
- [_M_detach_singular](#), 1028
- [_M_get_mutex](#), 1028
- [_M_invalidate_all](#), 1028
- [_M_invalidate_if](#), 1028
- [_M_iterators](#), 1029
- [_M_revalidate_singular](#), 1028
- [_M_transfer_from_if](#), 1029
- [_M_version](#), 1029
- [__gnu_debug:: Safe_iterator< _Iterator, _Sequence, _Category >](#), 1029
- [_M_attach](#), 1033
- [_M_attach_single](#), 1034
- [_M_attached_to](#), 1034
- [_M_before_dereferenceable](#), 1034
- [_M_can_compare](#), 1034
- [_M_dereferenceable](#), 1034
- [_M_detach](#), 1034
- [_M_detach_single](#), 1035
- [_M_get_mutex](#), 1035
- [_M_incrementable](#), 1035
- [_M_invalidate](#), 1035
- [_M_is_before_begin](#), 1035
- [_M_is_begin](#), 1035
- [_M_is_beginnest](#), 1035
- [_M_is_end](#), 1036
- [_M_next](#), 1038
- [_M_prior](#), 1038
- [_M_reset](#), 1036
- [_M_sequence](#), 1038
- [_M_singular](#), 1036
- [_M_unlink](#), 1036
- [_M_version](#), 1039
- [_S_constant](#), 1036
- [_Safe_iterator](#), 1032, 1033
- [base](#), 1036
- [operator _Iterator](#), 1037
- [operator*](#), 1037
- [operator++](#), 1037
- [operator->](#), 1037
- [operator=](#), 1038
- [__gnu_debug:: Safe_iterator_base](#), 1039
- [_M_attach](#), 1041
- [_M_attach_single](#), 1041
- [_M_attached_to](#), 1041
- [_M_can_compare](#), 1041
- [_M_detach](#), 1041
- [_M_detach_single](#), 1041
- [_M_get_mutex](#), 1041
- [_M_invalidate](#), 1042
- [_M_next](#), 1042
- [_M_prior](#), 1042
- [_M_reset](#), 1042
- [_M_sequence](#), 1042
- [_M_singular](#), 1042
- [_M_unlink](#), 1042
- [_M_version](#), 1043
- [_Safe_iterator_base](#), 1040
- [__gnu_debug:: Safe_local_iterator< _Iterator, _Sequence >](#), 1043
- [_M_attach](#), 1046
- [_M_attach_single](#), 1046, 1047
- [_M_attached_to](#), 1047
- [_M_can_compare](#), 1047
- [_M_dereferenceable](#), 1047
- [_M_detach](#), 1047
- [_M_detach_single](#), 1047
- [_M_get_mutex](#), 1047
- [_M_in_same_bucket](#), 1048
- [_M_incrementable](#), 1048
- [_M_invalidate](#), 1048
- [_M_is_begin](#), 1048
- [_M_is_end](#), 1048
- [_M_next](#), 1051
- [_M_prior](#), 1051
- [_M_reset](#), 1048
- [_M_sequence](#), 1051
- [_M_singular](#), 1048
- [_M_unlink](#), 1048
- [_M_version](#), 1051
- [_S_constant](#), 1049
- [_Safe_local_iterator](#), 1045, 1046
- [base](#), 1049
- [bucket](#), 1049
- [operator _Iterator](#), 1049
- [operator*](#), 1049
- [operator++](#), 1049, 1050
- [operator->](#), 1050
- [operator=](#), 1050
- [__gnu_debug:: Safe_local_iterator_base](#), 1051
- [_M_attach](#), 1053
- [_M_attach_single](#), 1053
- [_M_attached_to](#), 1053
- [_M_can_compare](#), 1054
- [_M_detach](#), 1054
- [_M_detach_single](#), 1054
- [_M_get_mutex](#), 1054
- [_M_invalidate](#), 1054
- [_M_next](#), 1055
- [_M_prior](#), 1055
- [_M_reset](#), 1054
- [_M_sequence](#), 1055
- [_M_singular](#), 1054
- [_M_unlink](#), 1054
- [_M_version](#), 1055
- [_Safe_local_iterator_base](#), 1053

- `__gnu_debug:: Safe_node_sequence< _Sequence >`, 1055
 - `_M_const_iterators`, 1058
 - `_M_detach_all`, 1057
 - `_M_detach_singular`, 1057
 - `_M_get_mutex`, 1057
 - `_M_invalidate_all`, 1057
 - `_M_invalidate_if`, 1057
 - `_M_iterators`, 1058
 - `_M_revalidate_singular`, 1057
 - `_M_swap`, 1057
 - `_M_transfer_from_if`, 1057
 - `_M_version`, 1058
- `__gnu_debug:: Safe_sequence< _Sequence >`, 1058
 - `_M_const_iterators`, 1061
 - `_M_detach_all`, 1060
 - `_M_detach_singular`, 1060
 - `_M_get_mutex`, 1060
 - `_M_invalidate_all`, 1060
 - `_M_invalidate_if`, 1060
 - `_M_iterators`, 1061
 - `_M_revalidate_singular`, 1060
 - `_M_swap`, 1060
 - `_M_transfer_from_if`, 1060
 - `_M_version`, 1061
- `__gnu_debug:: Safe_sequence_base`, 1061
 - `_M_const_iterators`, 1063
 - `_M_detach_all`, 1062
 - `_M_detach_singular`, 1062
 - `_M_get_mutex`, 1062
 - `_M_invalidate_all`, 1063
 - `_M_iterators`, 1063
 - `_M_revalidate_singular`, 1063
 - `_M_swap`, 1063
 - `_M_version`, 1063
 - `~_Safe_sequence_base`, 1062
- `__gnu_debug:: Safe_unordered_container< _Container >`, 1063
 - `_M_const_iterators`, 1066
 - `_M_const_local_iterators`, 1066
 - `_M_detach_all`, 1065
 - `_M_detach_singular`, 1065
 - `_M_get_mutex`, 1065
 - `_M_invalidate_all`, 1065
 - `_M_invalidate_if`, 1065
 - `_M_invalidate_local_if`, 1066
 - `_M_iterators`, 1066
 - `_M_local_iterators`, 1066
 - `_M_revalidate_singular`, 1066
 - `_M_swap`, 1066
 - `_M_version`, 1067
- `__gnu_debug:: Safe_unordered_container_base`, 1067
 - `_M_const_iterators`, 1069
 - `_M_const_local_iterators`, 1069
 - `_M_detach_all`, 1068
 - `_M_detach_singular`, 1068
 - `_M_get_mutex`, 1068
 - `_M_invalidate_all`, 1068
 - `_M_iterators`, 1069
 - `_M_local_iterators`, 1069
 - `_M_revalidate_singular`, 1069
 - `_M_swap`, 1069
 - `_M_version`, 1069
 - `~_Safe_unordered_container_base`, 1068
- `__gnu_debug:: Safe_vector< _SafeSequence, _BaseSequence >`, 1070
- `__gnu_debug:: Sequence_traits< _Sequence >`, 1073
- `__gnu_debug:: basic_string< _CharT, _Traits, _Allocator >`, 1556
 - `append`, 1561–1563
 - `assign`, 1563, 1565, 1566
 - `at`, 1566, 1568
 - `back`, 1568
 - `capacity`, 1569
 - `compare`, 1569–1571
 - `data`, 1571
 - `empty`, 1571
 - `erase`, 1572
 - `find`, 1572, 1573
 - `find_first_not_of`, 1573
 - `find_first_of`, 1574
 - `find_last_not_of`, 1575
 - `find_last_of`, 1575, 1576
 - `front`, 1576
 - `get_allocator`, 1577
 - `insert`, 1577–1580
 - `length`, 1581
 - `max_size`, 1581
 - `npos`, 1590
 - `operator __sv_type`, 1581
 - `operator+=`, 1581
 - `replace`, 1582–1587
 - `reserve`, 1588
 - `rfind`, 1588, 1589
 - `size`, 1589
 - `swap`, 1589
- `__gnu_internal`, 459
- `__gnu_parallel`, 460
 - `_AlgorithmStrategy`, 468
 - `_BinIndex`, 467
 - `_CASable`, 467
 - `_CASable_bits`, 508
 - `_CASable_mask`, 508
 - `_FindAlgorithm`, 468
 - `_MultiwayMergeAlgorithm`, 468
 - `_Parallelism`, 468
 - `_PartialSumAlgorithm`, 468
 - `_SequenceIndex`, 468

- `__SortAlgorithm`, 468
- `__SplittingAlgorithm`, 469
- `__ThreadIndex`, 468
- `__calc_borders`, 469
- `__compare_and_swap`, 469
- `__decode2`, 469
- `__determine_samples`, 470
- `__encode2`, 470
- `__equally_split`, 471
- `__equally_split_point`, 471
- `__fetch_and_add`, 471
- `__find_template`, 472–474
- `__for_each_template_random_access`, 474
- `__for_each_template_random_access_ed`, 475
- `__for_each_template_random_access_omp_loop`, 476
- `__for_each_template_random_access_omp_loop_static`, 476
- `__for_each_template_random_access_workstealing`, 477
- `__is_sorted`, 478
- `__median_of_three_iterators`, 478
- `__merge_advance`, 478
- `__merge_advance_movc`, 479
- `__merge_advance_usual`, 480
- `__parallel_merge_advance`, 480, 481
- `__parallel_nth_element`, 481
- `__parallel_partial_sort`, 483
- `__parallel_partial_sum`, 483
- `__parallel_partial_sum_basecase`, 483
- `__parallel_partial_sum_linear`, 484
- `__parallel_partition`, 484
- `__parallel_random_shuffle`, 485
- `__parallel_random_shuffle_drs`, 485
- `__parallel_random_shuffle_drs_pu`, 486
- `__parallel_sort`, 486–490
- `__parallel_sort_qs`, 491
- `__parallel_sort_qs_conquer`, 491
- `__parallel_sort_qs_divide`, 492
- `__parallel_sort_qsb`, 492
- `__parallel_unique_copy`, 492, 493
- `__qsb_conquer`, 493
- `__qsb_divide`, 494
- `__qsb_local_sort_with_helping`, 494
- `__random_number_pow2`, 495
- `__rd_log2`, 495
- `__round_up_to_pow2`, 495
- `__search_template`, 495
- `__sequential_multiway_merge`, 496
- `__sequential_random_shuffle`, 497
- `__shrink`, 497
- `__shrink_and_double`, 497
- `__yield`, 498
- `list_partition`, 498
- `max`, 498
- `min`, 499
- `multiseq_partition`, 499
- `multiseq_selection`, 499
- `multiway_merge`, 500
- `multiway_merge_3_variant`, 501
- `multiway_merge_4_variant`, 502
- `multiway_merge_exact_splitting`, 502
- `multiway_merge_loser_tree`, 503
- `multiway_merge_loser_tree_sentinel`, 503
- `multiway_merge_loser_tree_unguarded`, 504
- `multiway_merge_sampling_splitting`, 504
- `multiway_merge_sentinels`, 505
- `parallel_balanced`, 468
- `parallel_multiway_merge`, 506
- `parallel_omp_loop`, 468
- `parallel_omp_loop_static`, 468
- `parallel_sort_mwms`, 507
- `parallel_sort_mwms_pu`, 507
- `parallel_taskqueue`, 468
- `parallel_unbalanced`, 468
- `sequential`, 468
- `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>`, 947
 - `_M_bins_begin`, 948
 - `_M_num_threads`, 948
 - `_M_sd`, 948
 - `_M_seed`, 948
 - `_bins_end`, 948
- `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>`, 946
 - `_DRandomShufflingGlobalData`, 946
 - `_M_bin_proc`, 946
 - `_M_dist`, 946
 - `_M_num_bins`, 947
 - `_M_num_bits`, 947
 - `_M_source`, 947
 - `_M_starts`, 947
 - `_M_temporaries`, 947
- `__gnu_parallel::DummyReduct`, 948
- `__gnu_parallel::EqualFromLess<_T1, _T2, _Compare>`, 949
 - `first_argument_type`, 950
 - `result_type`, 950
 - `second_argument_type`, 950
- `__gnu_parallel::EqualTo<_T1, _T2>`, 950
 - `first_argument_type`, 951
 - `result_type`, 951
 - `second_argument_type`, 951
- `__gnu_parallel::GuardedIterator<_RAIter, _Compare>`, 959
 - `_GuardedIterator`, 960
 - `operator _RAIter`, 960
 - `operator<`, 961

operator<=, [961](#)
 operator*, [960](#)
 operator++, [960](#)
 __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [966](#)
 first, [969](#)
 first_type, [967](#)
 make_pair, [968](#)
 operator!=, [968](#)
 operator<, [968](#)
 operator<=, [968](#)
 operator>, [968](#)
 operator>=, [969](#)
 operator==, [968](#)
 second, [969](#)
 second_type, [967](#)
 swap, [967](#), [969](#)
 __gnu_parallel::_IteratorTriple< _Iterator1, _Iterator2, _Iterator3, _IteratorCategory >, [969](#)
 __gnu_parallel::_Job< _DifferenceTp >, [970](#)
 _M_first, [970](#)
 _M_last, [970](#)
 _M_load, [971](#)
 __gnu_parallel::_Less< _T1, _T2 >, [972](#)
 first_argument_type, [973](#)
 result_type, [973](#)
 second_argument_type, [973](#)
 __gnu_parallel::_Lexicographic< _T1, _T2, _Compare >, [973](#)
 first_argument_type, [974](#)
 result_type, [974](#)
 second_argument_type, [974](#)
 __gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare >, [974](#)
 first_argument_type, [975](#)
 result_type, [975](#)
 second_argument_type, [975](#)
 __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >, [983](#)
 _M_log_k, [984](#)
 __delete_min_insert, [983](#)
 __get_min_source, [983](#)
 __insert_start, [984](#)
 __gnu_parallel::_LoserTree< false, _Tp, _Compare >, [984](#)
 __delete_min_insert, [985](#)
 __get_min_source, [986](#)
 __init_winner, [986](#)
 __insert_start, [986](#)
 __gnu_parallel::_LoserTreeBase< _Tp, _Compare >, [986](#)
 _LoserTreeBase, [988](#)
 _M_comp, [989](#)
 _M_first_insert, [989](#)
 _M_log_k, [989](#)
 _M_losers, [989](#)
 __get_min_source, [988](#)
 __insert_start, [988](#)
 ~_LoserTreeBase, [988](#)
 __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser, [981](#)
 _M_key, [982](#)
 _M_source, [982](#)
 _M_sup, [982](#)
 __gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >, [989](#)
 __gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >, [990](#)
 __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >, [991](#)
 __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser, [982](#)
 __gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >, [992](#)
 __gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >, [993](#)
 __gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare >, [994](#)
 __gnu_parallel::_LoserTreeTraits< _Tp >, [994](#)
 _M_use_pointer, [995](#)
 __gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >, [995](#)
 __gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >, [996](#)
 __gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare >, [997](#)
 __gnu_parallel::_Multiplies< _Tp1, _Tp2, _Result >, [999](#)
 first_argument_type, [999](#)
 result_type, [999](#)
 second_argument_type, [999](#)
 __gnu_parallel::_Nothing, [1010](#)
 operator(), [1011](#)
 __gnu_parallel::_PMWMSSortingData< _RAIter >, [1014](#)
 _M_num_threads, [1015](#)
 _M_offsets, [1015](#)
 _M_pieces, [1015](#)
 _M_samples, [1015](#)
 _M_source, [1015](#)
 _M_starts, [1015](#)
 _M_temporary, [1015](#)
 __gnu_parallel::_Piece< _DifferenceTp >, [1012](#)
 _M_begin, [1013](#)
 _M_end, [1013](#)
 __gnu_parallel::_Plus< _Tp1, _Tp2, _Result >, [1013](#)
 first_argument_type, [1014](#)
 result_type, [1014](#)
 second_argument_type, [1014](#)
 __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >, [1018](#)

- `_PseudoSequence`, 1019
- `begin`, 1019
- `end`, 1019
- `__gnu_parallel::_PseudoSequenceIterator< _Tp, _DifferenceTp >`, 1019
- `__gnu_parallel::_QSBThreadLocal< _RAIter >`, 1020
 - `_M_elements_leftover`, 1021
 - `_M_global`, 1021
 - `_M_initial`, 1021
 - `_M_leftover_parts`, 1021
 - `_M_num_threads`, 1021
 - `_Piece`, 1020
 - `_QSBThreadLocal`, 1021
- `__gnu_parallel::_RandomNumber`, 1022
 - `_RandomNumber`, 1022
 - `__genrand_bits`, 1023
 - `operator()`, 1023
- `__gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >`, 1024
 - `_RestrictedBoundedConcurrentQueue`, 1025
 - `~_RestrictedBoundedConcurrentQueue`, 1025
 - `pop_back`, 1025
 - `pop_front`, 1025
 - `push_front`, 1025
- `__gnu_parallel::_SamplingSorter< __stable, _RAIter, _StrictWeakOrdering >`, 1070
- `__gnu_parallel::_SamplingSorter< false, _RAIter, _StrictWeakOrdering >`, 1071
- `__gnu_parallel::_Settings`, 1073
 - `accumulate_minimal_n`, 1075
 - `adjacent_difference_minimal_n`, 1075
 - `cache_line_size`, 1075
 - `count_minimal_n`, 1075
 - `fill_minimal_n`, 1075
 - `find_increasing_factor`, 1075
 - `find_initial_block_size`, 1075
 - `find_maximum_block_size`, 1075
 - `find_scale_factor`, 1075
 - `find_sequential_search_size`, 1075
 - `for_each_minimal_n`, 1076
 - `generate_minimal_n`, 1076
 - `get`, 1074
 - `L1_cache_size`, 1076
 - `L2_cache_size`, 1076
 - `max_element_minimal_n`, 1076
 - `merge_minimal_n`, 1076
 - `merge_oversampling`, 1076
 - `min_element_minimal_n`, 1076
 - `multiway_merge_minimal_k`, 1076
 - `multiway_merge_minimal_n`, 1076
 - `multiway_merge_oversampling`, 1076
 - `nth_element_minimal_n`, 1077
 - `partial_sort_minimal_n`, 1077
 - `partial_sum_dilation`, 1077
 - `partial_sum_minimal_n`, 1077
 - `partition_chunk_share`, 1077
 - `partition_chunk_size`, 1077
 - `partition_minimal_n`, 1077
 - `qsb_steals`, 1077
 - `random_shuffle_minimal_n`, 1077
 - `replace_minimal_n`, 1077
 - `search_minimal_n`, 1078
 - `set`, 1074
 - `set_difference_minimal_n`, 1078
 - `set_intersection_minimal_n`, 1078
 - `set_symmetric_difference_minimal_n`, 1078
 - `set_union_minimal_n`, 1078
 - `sort_minimal_n`, 1078
 - `sort_mwms_oversampling`, 1078
 - `sort_qs_num_samples_preset`, 1078
 - `sort_qsb_base_case_maximal_n`, 1078
 - `TLB_size`, 1078
 - `transform_minimal_n`, 1079
 - `unique_copy_minimal_n`, 1079
- `__gnu_parallel::_SplitConsistently< __exact, _RAIter, _Compare, _SortingPlacesIterator >`, 1080
- `__gnu_parallel::_SplitConsistently< false, _RAIter, _Compare, _SortingPlacesIterator >`, 1080
- `__gnu_parallel::_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIterator >`, 1080
- `__gnu_parallel::_accumulate_binop_reduct< _BinOp >`, 800
- `__gnu_parallel::_accumulate_selector< _It >`, 800
 - `_M_finish_iterator`, 802
 - `operator()`, 801
- `__gnu_parallel::_adjacent_difference_selector< _It >`, 802
 - `_M_finish_iterator`, 802
- `__gnu_parallel::_adjacent_find_selector`, 803
 - `_M_sequential_algorithm`, 803
- `__gnu_parallel::_adjacent_difference_selector< _It >`, 802
 - `_M_finish_iterator`, 802
- `__gnu_parallel::_binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`, 815
 - `argument_type`, 816
 - `result_type`, 816
- `__gnu_parallel::_binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`, 816
 - `argument_type`, 817
 - `result_type`, 817
- `__gnu_parallel::_count_if_selector< _It, _Diff >`, 821
 - `_M_finish_iterator`, 823
 - `operator()`, 822
- `__gnu_parallel::_count_selector< _It, _Diff >`, 823
 - `_M_finish_iterator`, 824
 - `operator()`, 823
- `__gnu_parallel::_fill_selector< _It >`, 840

[_M_finish_iterator](#), 841
[operator\(\)](#), 840
[__gnu_parallel::__find_first_of_selector< _FIterator >](#), 841
[_M_sequential_algorithm](#), 842
[operator\(\)](#), 842
[__gnu_parallel::__find_if_selector](#), 842
[_M_sequential_algorithm](#), 843
[operator\(\)](#), 843
[__gnu_parallel::__for_each_selector< _It >](#), 844
[_M_finish_iterator](#), 845
[operator\(\)](#), 845
[__gnu_parallel::__generate_selector< _It >](#), 845
[_M_finish_iterator](#), 847
[operator\(\)](#), 846
[__gnu_parallel::__generic_find_selector](#), 847
[__gnu_parallel::__generic_for_each_selector< _It >](#), 847
[_M_finish_iterator](#), 849
[__gnu_parallel::__identity_selector< _It >](#), 849
[_M_finish_iterator](#), 850
[operator\(\)](#), 850
[__gnu_parallel::__inner_product_selector< _It, _It2, _Tp >](#), 850
[_M_finish_iterator](#), 852
[__begin1_iterator](#), 852
[__begin2_iterator](#), 852
[__inner_product_selector](#), 851
[operator\(\)](#), 851
[__gnu_parallel::__max_element_reduct< _Compare, _It >](#), 853
[__gnu_parallel::__min_element_reduct< _Compare, _It >](#), 854
[__gnu_parallel::__mismatch_selector](#), 855
[_M_sequential_algorithm](#), 856
[operator\(\)](#), 856
[__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< _sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 859
[__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 859
[__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< _sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 860
[__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 860
[__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< _sentinels, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 860
[__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#), 861
[__gnu_parallel::__replace_if_selector< _It, _Op, _Tp >](#), 874
[_M_finish_iterator](#), 875
[__new_val](#), 875
[__replace_if_selector](#), 875
[operator\(\)](#), 875
[__gnu_parallel::__replace_selector< _It, _Tp >](#), 876
[_M_finish_iterator](#), 877
[__new_val](#), 877
[__replace_selector](#), 876
[operator\(\)](#), 877
[__gnu_parallel::__transform1_selector< _It >](#), 878
[_M_finish_iterator](#), 879
[operator\(\)](#), 878
[__gnu_parallel::__transform2_selector< _It >](#), 879
[_M_finish_iterator](#), 880
[operator\(\)](#), 879
[__gnu_parallel::__unary_negate< _Predicate, argument_type >](#), 880
[argument_type](#), 881
[result_type](#), 881
[__gnu_parallel::balanced_quicksort_tag](#), 1166
[__get_num_threads](#), 1167
[set_num_threads](#), 1167
[__gnu_parallel::balanced_tag](#), 1167
[__get_num_threads](#), 1168
[set_num_threads](#), 1168
[__gnu_parallel::constant_size_blocks_tag](#), 1865
[__gnu_parallel::default_parallel_tag](#), 1957
[__get_num_threads](#), 1958
[set_num_threads](#), 1958
[__gnu_parallel::equal_split_tag](#), 2044
[__gnu_parallel::exact_tag](#), 2051
[__get_num_threads](#), 2051
[set_num_threads](#), 2051
[__gnu_parallel::find_tag](#), 2066
[__gnu_parallel::growing_blocks_tag](#), 2129
[__gnu_parallel::multiway_mergesort_exact_tag](#), 2480
[__get_num_threads](#), 2481
[__gnu_parallel::multiway_mergesort_sampling_tag](#), 2482
[__get_num_threads](#), 2482
[__gnu_parallel::multiway_mergesort_tag](#), 2483
[__get_num_threads](#), 2483
[__gnu_parallel::omp_loop_static_tag](#), 2561
[__get_num_threads](#), 2562
[__gnu_parallel::omp_loop_tag](#), 2562
[__get_num_threads](#), 2563
[__gnu_parallel::parallel_tag](#), 2596
[__get_num_threads](#), 2598
[parallel_tag](#), 2598

- set_num_threads, 2598
- __gnu_parallel::quicksort_tag, 2653
 - __get_num_threads, 2654
 - set_num_threads, 2654
- __gnu_parallel::sampling_tag, 2731
 - __get_num_threads, 2732
 - set_num_threads, 2732
- __gnu_parallel::sequential_tag, 2740
- __gnu_parallel::unbalanced_tag, 2956
 - __get_num_threads, 2956
 - set_num_threads, 2956
- __gnu_pbds, 508
- __gnu_pbds::associative_tag, 1118
- __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >, 1168
- __gnu_pbds::basic_branch_tag, 1170
- __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >, 1240
- __gnu_pbds::basic_hash_tag, 1241
- __gnu_pbds::basic_invalidation_guarantee, 1283
- __gnu_pbds::binary_heap_tag, 1750
- __gnu_pbds::binomial_heap_tag, 1761
- __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, 1780
 - cc_hash_max_collision_check_resize_trigger, 1782
 - external_load_access, 1781
 - get_load, 1782
 - is_grow_needed, 1782
 - is_resize_needed, 1782
 - notify_cleared, 1782
 - notify_erase_search_collision, 1782
 - notify_erase_search_end, 1782
 - notify_erase_search_start, 1782
 - notify_erased, 1783
 - notify_externally_resized, 1783
 - notify_find_search_collision, 1783
 - notify_find_search_end, 1783
 - notify_find_search_start, 1783
 - notify_insert_search_collision, 1783
 - notify_insert_search_end, 1783
 - notify_insert_search_start, 1784
 - notify_inserted, 1784
 - notify_resized, 1784
 - set_load, 1784
- __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >, 1784
 - cc_hash_table, 1786–1788
- __gnu_pbds::cc_hash_tag, 1789
- __gnu_pbds::container_error, 1876
 - what, 1877
- __gnu_pbds::container_tag, 1877
- __gnu_pbds::container_traits< Cntnr >, 1877
 - erase_can_throw, 1878
 - order_preserving, 1878
 - reverse_iteration, 1878
 - split_join_can_throw, 1878
- __gnu_pbds::container_traits_base< _Tag >, 1879
- __gnu_pbds::container_traits_base< binary_heap_tag >, 1879
- __gnu_pbds::container_traits_base< binomial_heap_tag >, 1879
- __gnu_pbds::container_traits_base< cc_hash_tag >, 1879
- __gnu_pbds::container_traits_base< gp_hash_tag >, 1880
- __gnu_pbds::container_traits_base< list_update_tag >, 1880
- __gnu_pbds::container_traits_base< ov_tree_tag >, 1880
- __gnu_pbds::container_traits_base< pairing_heap_tag >, 1881
- __gnu_pbds::container_traits_base< pat_trie_tag >, 1881
- __gnu_pbds::container_traits_base< rb_tree_tag >, 1881
- __gnu_pbds::container_traits_base< rc_binomial_heap_tag >, 1881
- __gnu_pbds::container_traits_base< splay_tree_tag >, 1882
- __gnu_pbds::container_traits_base< thin_heap_tag >, 1882
- __gnu_pbds::detail::bin_search_tree_const_it< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >, 1728
- __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1730
 - const_reference, 1731
 - difference_type, 1731
 - get_l_child, 1732
 - get_metadata, 1732
 - get_r_child, 1732
 - iterator_category, 1731
 - metadata_const_reference, 1731
 - metadata_type, 1731
 - operator!=, 1732
 - operator*, 1732
 - operator==, 1732
 - reference, 1731
 - value_type, 1732
- __gnu_pbds::detail::bin_search_tree_it< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >, 1733
- __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1734
 - const_reference, 1735

- difference_type, 1735
- get_l_child, 1736
- get_metadata, 1736
- get_r_child, 1737
- iterator_category, 1736
- metadata_const_reference, 1736
- metadata_type, 1736
- operator!=, 1737
- operator*, 1737
- operator==, 1737
- reference, 1736
- value_type, 1736
- __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >, 1737
- node_const_iterator, 1738
- __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >, 1738
- node_const_iterator, 1739
- __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >, 1742
- __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1744
- binary_heap_const_iterator_, 1746
- const_pointer, 1745
- const_reference, 1745
- difference_type, 1745
- iterator_category, 1745
- operator!=, 1746
- operator*, 1746
- operator->, 1746
- operator==, 1747
- pointer, 1745
- reference, 1745
- value_type, 1745
- __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1747
- binary_heap_point_const_iterator_, 1749
- const_pointer, 1748
- const_reference, 1748
- difference_type, 1748
- iterator_category, 1748
- operator!=, 1749
- operator*, 1749
- operator->, 1749
- operator==, 1750
- pointer, 1748
- reference, 1749
- value_type, 1749
- __gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >, 1757
- __gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >, 1759
- __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >, 1778
- __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >, 1777
- __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >, 1789
- empty, 1792
- get_comb_hash_fn, 1792
- get_eq_fn, 1793
- get_hash_fn, 1793
- get_resize_policy, 1793
- __gnu_pbds::detail::cond_dealtor< Entry, _Alloc >, 1856
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI >, 1867
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >, 1869
- type, 1870
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >, 1870
- type, 1870
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >, 1870
- type, 1871
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI >, 1871
- type, 1871
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI >, 1871
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >, 1872
- type, 1872
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >, 1872
- type, 1873
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >, 1873
- type, 1873
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >, 1873
- type, 1874
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >, 1874
- type, 1874
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >, 1874
- type, 1875
- __gnu_pbds::detail::container_base_dispatch< Key,

[null_type, _Alloc, pat_trie_tag, Policy_Tl >, 1875](#)
[type, 1875](#)
[__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_Tl >, 1875](#)
[__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >, 1876](#)
[type, 1876](#)
[__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >, 1867](#)
[type, 1867](#)
[__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >, 1867](#)
[type, 1868](#)
[__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type >, 1868](#)
[type, 1868](#)
[__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type >, 1868](#)
[type, 1869](#)
[__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >, 1869](#)
[type, 1869](#)
[__gnu_pbds::detail::default_comb_hash_fn, 1954](#)
[type, 1954](#)
[__gnu_pbds::detail::default_eq_fn< Key >, 1956](#)
[type, 1957](#)
[__gnu_pbds::detail::default_hash_fn< Key >, 1957](#)
[type, 1957](#)
[__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >, 1959](#)
[type, 1959](#)
[__gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >, 1959](#)
[type, 1959](#)
[__gnu_pbds::detail::default_trie_access_traits< Key >, 1960](#)
[__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >, 1960](#)
[type, 1960](#)
[__gnu_pbds::detail::default_update_policy, 1961](#)
[type, 1961](#)
[__gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >, 2002](#)
[__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, No_Throw >, 2042](#)
[__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >, 2042](#)
[__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >::type, 2949](#)
[__gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, true >, 2042](#)
[type, 2042](#)
[__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, No_Throw >, 2043](#)
[__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, false >, 2043](#)
[__gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, true >, 2043](#)
[__gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >, 2044](#)
[__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, 2121](#)
[empty, 2123](#)
[get_comb_probe_fn, 2124](#)
[get_eq_fn, 2124](#)
[get_hash_fn, 2124](#)
[get_probe_fn, 2124, 2125](#)
[get_resize_policy, 2125](#)
[__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >, 2153](#)
[__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false >, 2153](#)
[__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true >, 2154](#)
[__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >, 2157](#)
[__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >, 2158](#)
[__gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >, 2263](#)
[__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, 2264](#)
[const_pointer, 2265](#)
[const_reference, 2265](#)
[difference_type, 2265](#)
[iterator_category, 2265](#)
[left_child_next_sibling_heap_const_iterator_, 2266](#)
[operator!=, 2266](#)
[operator*, 2267](#)
[operator->, 2267](#)
[operator==, 2267](#)
[pointer, 2266](#)
[reference, 2266](#)
[value_type, 2266](#)
[__gnu_pbds::detail::left_child_next_sibling_heap_node< _Value, _Metadata, _Alloc >, 2267](#)
[__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, 2268](#)
[const_pointer, 2269](#)

- const_reference, [2269](#)
- difference_type, [2269](#)
- iterator_category, [2269](#)
- left_child_next_sibling_heap_node_point_const_iterator, [2270](#)
- operator!=, [2270](#)
- operator*, [2270](#)
- operator->, [2270](#)
- operator==, [2271](#)
- pointer, [2269](#)
- reference, [2270](#)
- value_type, [2270](#)
- __gnu_pbds::detail::lu_counter_metadata< Size_Type >, [2326](#)
- __gnu_pbds::detail::lu_counter_policy_base< Size_Type >, [2328](#)
- __gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >, [2328](#)
- __gnu_pbds::detail::mask_based_range_hashing< Size_Type >, [2367](#)
- __gnu_pbds::detail::maybe_null_type< Key, Mapped, _Alloc, Store_Hash >, [2376](#)
- __gnu_pbds::detail::maybe_null_type< Key, null_type, _Alloc, Store_Hash >, [2376](#)
- __gnu_pbds::detail::mod_based_range_hashing< Size_Type >, [2393](#)
- __gnu_pbds::detail::no_throw_copies< Key, Mapped >, [2491](#)
- __gnu_pbds::detail::no_throw_copies< Key, null_type >, [2491](#)
- __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2576](#)
- node_begin, [2578](#)
- node_end, [2578](#)
- __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >, [1857](#)
- __gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc >, [2579](#)
- get_l_child, [2580](#)
- get_r_child, [2580](#)
- __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc >, [2580](#)
- get_l_child, [2582](#)
- get_r_child, [2582](#)
- operator*, [2582](#)
- __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >, [2593](#)
- __gnu_pbds::detail::pat_trie_base, [2609](#)
- node_type, [2610](#)
- __gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >, [939](#)
- __gnu_pbds::detail::pat_trie_base::_Head< _ATraits, Metadata >, [961](#)
- __gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >, [962](#)
- __gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator, [1859](#)
- __gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::iterator, [2256](#)
- __gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >, [964](#)
- __gnu_pbds::detail::pat_trie_base::_Leaf< _ATraits, Metadata >, [971](#)
- __gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >, [998](#)
- __gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >, [998](#)
- __gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >, [1000](#)
- __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >, [1001](#)
- get_child, [1002](#)
- get_metadata, [1002](#)
- metadata_const_reference, [1002](#)
- metadata_type, [1002](#)
- num_children, [1002](#)
- operator!=, [1003](#)
- operator*, [1003](#)
- operator==, [1003](#)
- valid_prefix, [1003](#)
- __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >, [1006](#)
- get_child, [1008](#)
- get_metadata, [1008](#)
- metadata_const_reference, [1008](#)
- metadata_type, [1008](#)
- num_children, [1008](#)
- operator!=, [1008](#)
- operator*, [1009](#)
- operator==, [1009](#)
- valid_prefix, [1009](#)
- __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >, [2610](#)
- node_begin, [2612](#)
- node_end, [2612](#), [2613](#)
- node_type, [2612](#)
- __gnu_pbds::detail::probe_fn_base< _Alloc >, [2644](#)
- __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash >, [2658](#)
- __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >, [2659](#)
- __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >, [2659](#)
- __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false >, [2660](#)

- __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true >, [2660](#)
- __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >, [2661](#)
- __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >, [2661](#)
- __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >, [2662](#)
- __gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >, [2663](#)
- __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2676](#)
node_begin, [2679](#)
node_end, [2679](#)
- __gnu_pbds::detail::rb_tree_node< Value_Type, Metadata, _Alloc >, [2680](#)
- __gnu_pbds::detail::rc< _Node, _Alloc >, [2681](#)
- __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >, [2682](#)
- __gnu_pbds::detail::rebind_traits< _Alloc, T >, [2684](#)
- __gnu_pbds::detail::resize_policy< _Tp >, [2707](#)
- __gnu_pbds::detail::select_value_type< Key, Mapped >, [2737](#)
- __gnu_pbds::detail::select_value_type< Key, null_type >, [2737](#)
- __gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2796](#)
node_begin, [2799](#)
node_end, [2799](#)
- __gnu_pbds::detail::splay_tree_node< Value_Type, Metadata, _Alloc >, [2800](#)
- __gnu_pbds::detail::stored_data< _Tv, _Th, Store_Hash >, [2842](#)
- __gnu_pbds::detail::stored_data< _Tv, _Th, false >, [2842](#)
- __gnu_pbds::detail::stored_hash< _Th >, [2843](#)
- __gnu_pbds::detail::stored_value< _Tv >, [2844](#)
- __gnu_pbds::detail::synth_access_traits< Type_Traits, Set, ATraits >, [2865](#)
- __gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >, [2869](#)
- __gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >, [2915](#)
- __gnu_pbds::detail::tree_metadata_helper< Node_Update, false >, [2915](#)
- __gnu_pbds::detail::tree_metadata_helper< Node_Update, true >, [2915](#)
- __gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >, [2915](#)
- __gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc >, [2918](#)
- __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [2918](#)
node_const_iterator, [2919](#)
- __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [2919](#)
node_const_iterator, [2920](#)
- __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [2920](#)
node_const_iterator, [2922](#)
- __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [2922](#)
node_const_iterator, [2922](#)
- __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [2922](#)
node_const_iterator, [2923](#)
- __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [2924](#)
node_const_iterator, [2925](#)
- __gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >, [2927](#)
- __gnu_pbds::detail::trie_metadata_helper< Node_Update, false >, [2927](#)
- __gnu_pbds::detail::trie_metadata_helper< Node_Update, true >, [2928](#)
- __gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >, [2928](#)
- __gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, ATraits, _Alloc >, [2931](#)
end, [2932](#)
- __gnu_pbds::detail::trie_traits< Key, Data, ATraits, Node_Update, Tag, _Alloc >, [2938](#)
- __gnu_pbds::detail::trie_traits< Key, Mapped, ATraits, Node_Update, pat_trie_tag, _Alloc >, [2938](#)
node_const_iterator, [2939](#)
node_update, [2939](#)
synth_access_traits, [2939](#)
- __gnu_pbds::detail::trie_traits< Key, null_type, ATraits, Node_Update, pat_trie_tag, _Alloc >, [2940](#)
node_const_iterator, [2940](#)
node_update, [2940](#)
synth_access_traits, [2940](#)
- __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >, [2951](#)
- __gnu_pbds::direct_mask_range_hashing< Size_Type >, [1987](#)
operator(), [1988](#)
- __gnu_pbds::direct_mod_range_hashing< Size_Type >, [1988](#)
operator(), [1989](#)
- __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >, [2115](#)
gp_hash_table, [2116–2119](#)
- __gnu_pbds::gp_hash_tag, [2120](#)

- __gnu_pbds::hash_exponential_size_policy< Size_Type
>, 2154
- hash_exponential_size_policy, 2155
- __gnu_pbds::hash_load_check_resize_trigger< Exter-
nal_Load_Access, Size_Type >, 2155
- external_load_access, 2156
- get_loads, 2157
- hash_load_check_resize_trigger, 2156
- notify_cleared, 2157
- notify_inserted, 2157
- notify_resized, 2157
- set_loads, 2157
- __gnu_pbds::hash_prime_size_policy, 2163
- hash_prime_size_policy, 2163
- size_type, 2163
- __gnu_pbds::hash_standard_resize_policy< Size_Policy,
Trigger_Policy, External_Size_Access, Size_Type
>, 2165
- external_load_access, 2167
- get_actual_size, 2167
- get_loads, 2167
- get_new_size, 2167
- get_size_policy, 2168
- get_trigger_policy, 2168
- hash_standard_resize_policy, 2167
- resize, 2168
- set_loads, 2168
- __gnu_pbds::insert_error, 2180
- what, 2181
- __gnu_pbds::join_error, 2261
- what, 2261
- __gnu_pbds::linear_probe_fn< Size_Type >, 2281
- operator(), 2282
- __gnu_pbds::list_update< Key, Mapped, Eq_Fn, Up-
date_Policy, _Alloc >, 2306
- list_update, 2307
- __gnu_pbds::list_update_tag, 2307
- __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >,
2326
- max_count, 2327
- metadata_reference, 2327
- metadata_type, 2327
- operator(), 2327
- __gnu_pbds::lu_move_to_front_policy< _Alloc >, 2331
- metadata_reference, 2331
- metadata_type, 2331
- operator(), 2331
- __gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3,
_Tp4 >, 2497
- __gnu_pbds::null_type, 2498
- __gnu_pbds::ov_tree_tag, 2582
- __gnu_pbds::pairing_heap_tag, 2595
- __gnu_pbds::pat_trie_tag, 2613
- __gnu_pbds::point_invalidation_guarantee, 2628
- __gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc
>, 2638
- priority_queue, 2640
- __gnu_pbds::priority_queue_tag, 2644
- __gnu_pbds::quadratic_probe_fn< Size_Type >, 2650
- operator(), 2650
- __gnu_pbds::range_invalidation_guarantee, 2657
- __gnu_pbds::rb_tree_tag, 2680
- __gnu_pbds::rc_binomial_heap_tag, 2684
- __gnu_pbds::resize_error, 2706
- what, 2706
- __gnu_pbds::sample_probe_fn, 2718
- operator(), 2719
- sample_probe_fn, 2719
- swap, 2719
- __gnu_pbds::sample_range_hashing, 2719
- notify_resized, 2720
- operator(), 2720
- sample_range_hashing, 2720
- size_type, 2720
- swap, 2720
- __gnu_pbds::sample_ranged_hash_fn, 2720
- notify_resized, 2721
- operator(), 2721
- sample_ranged_hash_fn, 2721
- swap, 2721
- __gnu_pbds::sample_ranged_probe_fn, 2721
- __gnu_pbds::sample_resize_policy, 2722
- get_new_size, 2723
- is_resize_needed, 2723
- notify_cleared, 2723
- notify_erase_search_collision, 2723
- notify_erase_search_end, 2723
- notify_erase_search_start, 2723
- notify_erased, 2723
- notify_find_search_collision, 2723
- notify_find_search_end, 2723
- notify_find_search_start, 2723
- notify_insert_search_collision, 2724
- notify_insert_search_end, 2724
- notify_insert_search_start, 2724
- notify_inserted, 2724
- notify_resized, 2724
- sample_range_hashing, 2724
- sample_resize_policy, 2723
- size_type, 2722
- swap, 2724
- __gnu_pbds::sample_resize_trigger, 2724
- is_grow_needed, 2725
- is_resize_needed, 2725
- notify_cleared, 2725
- notify_erase_search_collision, 2726
- notify_erase_search_end, 2726
- notify_erase_search_start, 2726

- notify_erased, 2726
- notify_externally_resized, 2726
- notify_find_search_collision, 2726
- notify_find_search_end, 2726
- notify_find_search_start, 2726
- notify_insert_search_collision, 2726
- notify_insert_search_end, 2726
- notify_insert_search_start, 2726
- notify_inserted, 2727
- notify_resized, 2727
- sample_range_hashing, 2727
- sample_resize_trigger, 2725
- size_type, 2725
- swap, 2727
- __gnu_pbds::sample_size_policy, 2727
 - get_nearest_larger_size, 2728
 - get_nearest_smaller_size, 2728
 - sample_range_hashing, 2728
 - sample_size_policy, 2728
 - size_type, 2728
 - swap, 2728
- __gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >, 2728
- __gnu_pbds::sample_trie_access_traits, 2729
 - begin, 2729
 - e_pos, 2729
 - e_type, 2729
 - end, 2729
- __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2729
 - operator(), 2730
 - sample_trie_node_update, 2730
- __gnu_pbds::sample_update_policy, 2730
 - metadata_type, 2731
 - operator(), 2731
 - sample_update_policy, 2731
 - swap, 2731
- __gnu_pbds::sequence_tag, 2740
- __gnu_pbds::splay_tree_tag, 2800
- __gnu_pbds::string_tag, 2845
- __gnu_pbds::thin_heap_tag, 2871
- __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >, 2912
 - cmp_fn, 2914
 - tree, 2914
- __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >, 2916
 - find_by_order, 2917
 - operator(), 2917
 - order_of_key, 2917
- __gnu_pbds::tree_tag, 2918
- __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >, 2925
 - access_traits, 2926
 - trie, 2926, 2927
 - __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2928
 - find_by_order, 2930
 - operator(), 2930
 - order_of_key, 2930
 - order_of_prefix, 2930
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2932
 - a_const_iterator, 2934
 - access_traits, 2934
 - allocator_type, 2934
 - operator(), 2934
 - prefix_range, 2934, 2935
 - size_type, 2934
 - __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >, 2935
 - begin, 2936
 - const_iterator, 2936
 - e_pos, 2936
 - e_type, 2936
 - end, 2937
 - __gnu_pbds::trie_tag, 2937
 - __gnu_pbds::trivial_iterator_tag, 2941
 - __gnu_sequential, 510
 - __heap_select
 - std, 636
 - __init_winner
 - __gnu_parallel::_LoserTree< false, _Tp, _Compare >, 986
 - __inner_product_selector
 - __gnu_parallel::_inner_product_selector< _It, _It2, _Tp >, 851
 - __inplace_stable_sort
 - std, 637
 - __insert_start
 - __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >, 984
 - __gnu_parallel::_LoserTree< false, _Tp, _Compare >, 986
 - __gnu_parallel::_LoserTreeBase< _Tp, _Compare >, 988
 - __insertion_sort
 - std, 637
 - __int_traits
 - __gnu_cxx, 439
 - __introsort_loop
 - std, 637
 - __invoke
 - Utilities, 341
 - __ioint
 - std, 710
 - __is_sorted
 - __gnu_parallel, 478

- __iterator_category
 - Iterators, 177
- __lg
 - std, 637
- __median
 - SGL, 157, 158
- __median_of_three_iterators
 - __gnu_parallel, 478
- __merge_adaptive
 - std, 637
- __merge_advance
 - __gnu_parallel, 478
- __merge_advance_movc
 - __gnu_parallel, 479
- __merge_advance_usual
 - __gnu_parallel, 480
- __merge_without_buffer
 - std, 638
- __move_median_to_first
 - std, 638
- __move_merge
 - std, 638
- __move_merge_adaptive
 - std, 638
- __move_merge_adaptive_backward
 - std, 639
- __multiline
 - std::regex_constants, 780
- __new_val
 - __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >, 875
 - __gnu_parallel::__replace_selector< _It, _Tp >, 877
- __num_bitmaps
 - __gnu_cxx::__detail, 450
- __num_blocks
 - __gnu_cxx::__detail, 450
- __num_get_type
 - std::basic_ios< _CharT, _Traits >, 1287
 - std::basic_ofstream< _CharT, _Traits >, 1436
 - std::basic_ostream< _CharT, _Traits >, 1470
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1502
- __num_put_type
 - std::basic_fstream< _CharT, _Traits >, 1197
 - std::basic_ifstream< _CharT, _Traits >, 1248
 - std::basic_ios< _CharT, _Traits >, 1288
 - std::basic_iostream< _CharT, _Traits >, 1312
 - std::basic_istream< _CharT, _Traits >, 1357
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1395
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1684
- __parallel_merge_advance
 - __gnu_parallel, 480, 481
- __parallel_nth_element
 - __gnu_parallel, 481
- __parallel_partial_sort
 - __gnu_parallel, 483
- __parallel_partial_sum
 - __gnu_parallel, 483
- __parallel_partial_sum_basecase
 - __gnu_parallel, 483
- __parallel_partial_sum_linear
 - __gnu_parallel, 484
- __parallel_partition
 - __gnu_parallel, 484
- __parallel_random_shuffle
 - __gnu_parallel, 485
- __parallel_random_shuffle_drs
 - __gnu_parallel, 485
- __parallel_random_shuffle_drs_pu
 - __gnu_parallel, 486
- __parallel_sort
 - __gnu_parallel, 486–490
- __parallel_sort_qs
 - __gnu_parallel, 491
- __parallel_sort_qs_conquer
 - __gnu_parallel, 491
- __parallel_sort_qs_divide
 - __gnu_parallel, 492
- __parallel_sort_qsb
 - __gnu_parallel, 492
- __parallel_unique_copy
 - __gnu_parallel, 492, 493
- __partition
 - std, 639
- __polynomial
 - std::regex_constants, 780
- __ptr_rebind
 - std, 632
- __qsb_conquer
 - __gnu_parallel, 493
- __qsb_divide
 - __gnu_parallel, 494
- __qsb_local_sort_with_helping
 - __gnu_parallel, 494
- __random_number_pow2
 - __gnu_parallel, 495
- __rd_log2
 - __gnu_parallel, 495
- __regex_replace
 - Regular Expressions, 273
- __replace_if_selector
 - __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >, 875
- __replace_selector
 - __gnu_parallel::__replace_selector< _It, _Tp >, 876
- __reverse

- std, 639
- __rotate
 - std, 640
- __rotate_adaptive
 - std, 640
- __round_up_to_pow2
 - __gnu_parallel, 495
- __sample
 - std, 641
- __search_n_aux
 - std, 641
- __search_template
 - __gnu_parallel, 495
- __sequential_multiway_merge
 - __gnu_parallel, 496
- __sequential_random_shuffle
 - __gnu_parallel, 497
- __shrink
 - __gnu_parallel, 497
- __shrink_and_double
 - __gnu_parallel, 497
- __stable_partition_adaptive
 - std, 641
- __static_pointer_cast
 - __gnu_cxx, 439
- __streambuf_type
 - std::basic_streambuf< _CharT, _Traits >, 1542
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3127
- __umap_traits
 - std, 632
- __ummap_traits
 - std, 632
- __umset_traits
 - std, 632
- __unguarded_insertion_sort
 - std, 642
- __unguarded_linear_insert
 - std, 642
- __unguarded_partition
 - std, 642
- __unguarded_partition_pivot
 - std, 642
- __unique_copy
 - std, 643
- __uset_traits
 - std, 633
- __valid_range
 - __gnu_debug, 458, 459
- __valid_range_aux
 - __gnu_debug, 459
- __verbose_terminate_handler
 - Exceptions, 137
- __versa_string
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 884–887
- __yield
 - __gnu_parallel, 498
- ~_LoserTreeBase
 - __gnu_parallel::LoserTreeBase< _Tp, _Compare >, 988
- ~_RestrictedBoundedConcurrentQueue
 - __gnu_parallel::RestrictedBoundedConcurrentQueue< _Tp >, 1025
- ~_Safe_sequence_base
 - __gnu_debug::Safe_sequence_base, 1062
- ~_Safe_unordered_container_base
 - __gnu_debug::Safe_unordered_container_base, 1068
- ~__versa_string
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 887
- ~any
 - std::any, 1113
 - std::experimental::fundamentals_v1::any, 1116
- ~auto_ptr
 - std::auto_ptr< _Tp >, 1153
- ~basic_filebuf
 - std::basic_filebuf< _CharT, _Traits >, 1174
- ~basic_fstream
 - std::basic_fstream< _CharT, _Traits >, 1200
- ~basic_ifstream
 - std::basic_ifstream< _CharT, _Traits >, 1251
- ~basic_ios
 - std::basic_ios< _CharT, _Traits >, 1290
- ~basic_iostream
 - std::basic_iostream< _CharT, _Traits >, 1314
- ~basic_istream
 - std::basic_istream< _CharT, _Traits >, 1359
- ~basic_istreamstream
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1398
- ~basic_ofstream
 - std::basic_ofstream< _CharT, _Traits >, 1440
- ~basic_ostream
 - std::basic_ostream< _CharT, _Traits >, 1472
- ~basic_ostringstream
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1504
- ~basic_regex
 - std::basic_regex< _Ch_type, _Rx_traits >, 1533
- ~basic_streambuf
 - std::basic_streambuf< _CharT, _Traits >, 1543
- ~basic_string
 - std::basic_string< _CharT, _Traits, _Alloc >, 1600
- ~basic_stringstream
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1687

- ~collate
 - std::collate< _CharT >, [1840](#)
- ~ctype
 - std::ctype< char >, [1901](#)
 - std::ctype< wchar_t >, [1912](#)
- ~deque
 - std::deque< _Tp, _Alloc >, [1971](#)
- ~facet
 - std::locale::facet, [2062](#)
- ~forward_list
 - std::forward_list< _Tp, _Alloc >, [2079](#)
- ~gslice
 - Numeric Arrays, [230](#)
- ~ios_base
 - std::ios_base, [2192](#)
- ~list
 - std::list< _Tp, _Alloc >, [2290](#)
- ~locale
 - std::locale, [2312](#)
- ~map
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2342](#)
- ~match_results
 - std::match_results< _Bi_iter, _Alloc >, [2370](#)
- ~messages
 - std::messages< _CharT >, [2388](#)
- ~money_get
 - std::money_get< _CharT, _InIter >, [2398](#)
- ~money_put
 - std::money_put< _CharT, _OutIter >, [2403](#)
- ~moneypunct
 - std::moneypunct< _CharT, _Intl >, [2408](#)
- ~multimap
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2437](#)
- ~multiset
 - std::multiset< _Key, _Compare, _Alloc >, [2464](#)
- ~num_get
 - std::num_get< _CharT, _InIter >, [2502](#)
- ~num_put
 - std::num_put< _CharT, _OutIter >, [2518](#)
- ~numpunct
 - std::numpunct< _CharT >, [2554](#)
- ~sentry
 - std::basic_ostream< _CharT, _Traits >::sentry, [2739](#)
- ~set
 - std::set< _Key, _Compare, _Alloc >, [2751](#)
- ~stdio_filebuf
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2807](#)
- ~temporary_buffer
 - __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, [2868](#)
- ~time_get
 - std::time_get< _CharT, _InIter >, [2884](#)
- ~time_put
 - std::time_put< _CharT, _OutIter >, [2905](#)
- ~type_info
 - std::type_info, [2951](#)
- ~unique_ptr
 - std::unique_ptr< _Tp, _Dp >, [2967](#)
 - std::unique_ptr< _Tp[], _Dp >, [2972](#)
- ~vector
 - std::vector< _Tp, _Alloc >, [3108](#)
- a
 - std::extreme_value_distribution< _RealType >, [2058](#)
 - std::weibull_distribution< _RealType >, [3143](#)
- a_const_iterator
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, [2934](#)
- abi, [510](#)
- abs
 - Complex Numbers, [189](#)
 - Time, [414](#)
- access_traits
 - __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >, [2926](#)
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, [2934](#)
- accumulate
 - Generalized Numeric operations, [6](#)
- accumulate_minimal_n
 - __gnu_parallel::Settings, [1075](#)
- acos
 - std, [644](#)
- acosh
 - std, [644](#)
- Adaptors for pointers to functions, [353](#)
 - ptr_fun, [353](#), [354](#)
- Adaptors for pointers to members, [354](#)
- add_const_t
 - Metaprogramming, [397](#)
- add_cv_t
 - Metaprogramming, [397](#)
- add_lvalue_reference_t
 - Metaprogramming, [397](#)
- add_pointer_t
 - Metaprogramming, [397](#)
- add_rvalue_reference_t
 - Metaprogramming, [397](#)
- add_volatile_t
 - Metaprogramming, [397](#)
- addressof
 - Utilities, [341](#)
- adjacent_difference
 - Generalized Numeric operations, [7](#)
- adjacent_difference_minimal_n
 - __gnu_parallel::Settings, [1075](#)
- adjacent_find

- Non-Mutating, [45](#)
- adjustfield
 - std::basic_fstream< _CharT, _Traits >, [1235](#)
 - std::basic_ifstream< _CharT, _Traits >, [1279](#)
 - std::basic_ios< _CharT, _Traits >, [1302](#)
 - std::basic_iostream< _CharT, _Traits >, [1348](#)
 - std::basic_istream< _CharT, _Traits >, [1385](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1427](#)
 - std::basic_ofstream< _CharT, _Traits >, [1461](#)
 - std::basic_ostream< _CharT, _Traits >, [1493](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1525](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1721](#)
 - std::ios_base, [2197](#)
- adopt_lock
 - Mutexes, [125](#)
- advance
 - std, [644](#)
- airy_ai
 - Mathematical Special Functions, [201](#)
- airy_aif
 - Mathematical Special Functions, [201](#)
- airy_ail
 - Mathematical Special Functions, [201](#)
- airy_bi
 - Mathematical Special Functions, [201](#)
- airy_bif
 - Mathematical Special Functions, [201](#)
- airy_bil
 - Mathematical Special Functions, [201](#)
- algo.h, [3496](#)
- algbase.h, [3506](#)
- algorithm, [3155](#), [3157](#)
- algorithmfwd.h, [3168](#), [3174](#)
- Algorithms, [4](#)
- align
 - Memory, [362](#)
- align.h, [3182](#)
- aligned_buffer.h, [3398](#)
- aligned_storage_t
 - Metaprogramming, [397](#)
- alignment_value
 - Metaprogramming, [401](#)
- all
 - std::bitset< _Nb >, [1769](#)
 - std::locale, [2315](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2009](#)
- all_of
 - Non-Mutating, [45](#)
- alloc_traits.h, [3182](#), [3183](#)
- allocate
 - __gnu_cxx::__alloc_traits< _Alloc, typename >, [806](#), [807](#)
 - std::allocator_traits< _Alloc >, [1095](#), [1096](#)
 - std::allocator_traits< allocator< _Tp > >, [1100](#)
 - std::allocator_traits< allocator< void > >, [1104](#)
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, [1108](#)
- allocate_shared
 - Pointer Abstractions, [373](#)
- allocated_ptr.h, [3183](#)
- allocator.h, [3184](#)
- allocator_type
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, ATraits, _Alloc >, [2934](#)
 - std::allocator_traits< _Alloc >, [1094](#)
 - std::allocator_traits< allocator< _Tp > >, [1099](#)
 - std::allocator_traits< allocator< void > >, [1103](#)
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, [1107](#)
 - std::set< _Key, _Compare, _Alloc >, [2746](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2983](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3015](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3042](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- Allocators, [367](#)
 - __allocator_base, [368](#)
- alpha
 - std::gamma_distribution< _RealType >, [2110](#)
- any, [3157](#), [3158](#)
 - std::any, [1112](#), [1113](#)
 - std::bitset< _Nb >, [1770](#)
 - std::experimental::fundamentals_v1::any, [1115](#), [1116](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2009](#)
- any_cast
 - Type-safe container of any type, [317](#)–[319](#)
 - Utilities, [341](#)–[343](#)
- any_of
 - Non-Mutating, [46](#)
- app
 - std::basic_fstream< _CharT, _Traits >, [1235](#)
 - std::basic_ifstream< _CharT, _Traits >, [1279](#)
 - std::basic_ios< _CharT, _Traits >, [1302](#)
 - std::basic_iostream< _CharT, _Traits >, [1348](#)
 - std::basic_istream< _CharT, _Traits >, [1385](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1427](#)
 - std::basic_ofstream< _CharT, _Traits >, [1461](#)
 - std::basic_ostream< _CharT, _Traits >, [1493](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1525](#)

- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1721
- std::ios_base, 2197
- append
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 888–890
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1561–1563
 - std::basic_string< _CharT, _Traits, _Alloc >, 1601–1604
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2009
- apply
 - Numeric Arrays, 230, 231
- apply_generator
 - __gnu_cxx::typelist, 451
- arg
 - Complex Numbers, 189
 - std, 645
- argument_type
 - __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >, 1740
 - __gnu_cxx::select1st< _Pair >, 2736
 - __gnu_cxx::select2nd< _Pair >, 2736
 - __gnu_cxx::subtractive_rng, 2863
 - __gnu_cxx::unary_compose< _Operation1, _Operation2 >, 2953
 - __gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >, 816
 - __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >, 817
 - __gnu_parallel::__unary_negate< _Predicate, argument_type >, 881
 - std::binder1st< _Operation >, 1752
 - std::binder2nd< _Operation >, 1754
 - std::const_mem_fun_ref_t< _Ret, _Tp >, 1863
 - std::const_mem_fun_t< _Ret, _Tp >, 1864
 - std::hash< __gnu_cxx::throw_value_limit >, 2138
 - std::hash< __gnu_cxx::throw_value_random >, 2138
 - std::logical_not< _Tp >, 2321
 - std::mem_fun_ref_t< _Ret, _Tp >, 2380
 - std::mem_fun_t< _Ret, _Tp >, 2381
 - std::negate< _Tp >, 2485
 - std::pointer_to_unary_function< _Arg, _Result >, 2631
 - std::unary_function< _Arg, _Result >, 2954
 - std::unary_negate< _Predicate >, 2955
- Arithmetic Function Object Classes, 355
- array, 3158, 3160
- Array creation functions, 310
 - make_array, 311
 - to_array, 311
- asin
 - std, 645
- asinh
 - std, 645
- assertions.h, 3368
- assign
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 890–893
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1563, 1565, 1566
 - std::basic_regex< _Ch_type, _Rx_traits >, 1534–1536
 - std::basic_string< _CharT, _Traits, _Alloc >, 1604–1608
 - std::deque< _Tp, _Alloc >, 1974, 1975
 - std::error_condition, 2050
 - std::forward_list< _Tp, _Alloc >, 2079, 2080
 - std::list< _Tp, _Alloc >, 2291, 2292
 - std::vector< _Tp, _Alloc >, 3108, 3109
- assoc_container.hpp, 3403
- assoc_laguerre
 - Mathematical Special Functions, 202
 - TR1 Mathematical Special Functions, 263
- assoc_laguerref
 - Mathematical Special Functions, 202
- assoc_laguerrel
 - Mathematical Special Functions, 203
- assoc_legendre
 - Mathematical Special Functions, 203
 - TR1 Mathematical Special Functions, 263
- assoc_legendref
 - Mathematical Special Functions, 204
- assoc_legendrel
 - Mathematical Special Functions, 204
- Associative, 127
- assume_aligned
 - Memory, 362
- async
 - Futures, 121
- at
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 894
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1566, 1568
 - std::basic_string< _CharT, _Traits, _Alloc >, 1608, 1609
 - std::deque< _Tp, _Alloc >, 1975, 1976
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2343
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2987
 - std::vector< _Tp, _Alloc >, 3109, 3110
- atan
- std, 645
- atanh

- std, [645](#)
- ate
 - std::basic_fstream< _CharT, _Traits >, [1236](#)
 - std::basic_ifstream< _CharT, _Traits >, [1279](#)
 - std::basic_ios< _CharT, _Traits >, [1302](#)
 - std::basic_iostream< _CharT, _Traits >, [1348](#)
 - std::basic_istream< _CharT, _Traits >, [1385](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1427](#)
 - std::basic_ofstream< _CharT, _Traits >, [1461](#)
 - std::basic_ostream< _CharT, _Traits >, [1493](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1525](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1721](#)
 - std::ios_base, [2198](#)
- atomic, [3160](#)
- atomic_base.h, [3184](#)
- atomic_bool
 - Atomics, [110](#)
- ATOMIC_BOOL_LOCK_FREE
 - Atomics, [110](#)
- atomic_char
 - Atomics, [110](#)
- atomic_char16_t
 - Atomics, [111](#)
- atomic_char32_t
 - Atomics, [111](#)
- atomic_compare_exchange_strong_explicit
 - Pointer Abstractions, [374](#)
- atomic_exchange_explicit
 - Pointer Abstractions, [374](#)
- atomic_futex.h, [3186](#)
- atomic_int
 - Atomics, [111](#)
- atomic_int16_t
 - Atomics, [111](#)
- atomic_int32_t
 - Atomics, [111](#)
- atomic_int64_t
 - Atomics, [111](#)
- atomic_int8_t
 - Atomics, [111](#)
- atomic_int_fast16_t
 - Atomics, [112](#)
- atomic_int_fast32_t
 - Atomics, [112](#)
- atomic_int_fast64_t
 - Atomics, [112](#)
- atomic_int_fast8_t
 - Atomics, [112](#)
- atomic_int_least16_t
 - Atomics, [112](#)
- atomic_int_least32_t
 - Atomics, [112](#)
- atomic_int_least64_t
 - Atomics, [112](#)
- atomic_int_least8_t
 - Atomics, [113](#)
- atomic_intmax_t
 - Atomics, [113](#)
- atomic_intptr_t
 - Atomics, [113](#)
- atomic_is_lock_free
 - Pointer Abstractions, [375](#)
- atomic_llong
 - Atomics, [113](#)
- atomic_load_explicit
 - Pointer Abstractions, [375](#)
- atomic_lockfree_defines.h, [3186](#)
- atomic_long
 - Atomics, [113](#)
- atomic_ptrdiff_t
 - Atomics, [113](#)
- atomic_schar
 - Atomics, [113](#)
- atomic_short
 - Atomics, [114](#)
- atomic_size_t
 - Atomics, [114](#)
- atomic_store_explicit
 - Pointer Abstractions, [375](#)
- atomic_timed_wait.h, [3187](#)
- atomic_uchar
 - Atomics, [114](#)
- atomic_uint
 - Atomics, [114](#)
- atomic_uint16_t
 - Atomics, [114](#)
- atomic_uint32_t
 - Atomics, [114](#)
- atomic_uint64_t
 - Atomics, [114](#)
- atomic_uint8_t
 - Atomics, [115](#)
- atomic_uint_fast16_t
 - Atomics, [115](#)
- atomic_uint_fast32_t
 - Atomics, [115](#)
- atomic_uint_fast64_t
 - Atomics, [115](#)
- atomic_uint_fast8_t
 - Atomics, [115](#)
- atomic_uint_least16_t
 - Atomics, [115](#)
- atomic_uint_least32_t
 - Atomics, [115](#)
- atomic_uint_least64_t

- Atomics, 116
- atomic_uint_least8_t
 - Atomics, 116
- atomic_uintmax_t
 - Atomics, 116
- atomic_uintptr_t
 - Atomics, 116
- atomic_ullong
 - Atomics, 116
- atomic_ulong
 - Atomics, 116
- atomic_ushort
 - Atomics, 116
- atomic_wait.h, 3188
- atomic_wchar_t
 - Atomics, 117
- atomic_word.h, 3582
- atomicity.h, 3398
- Atomics, 105
 - atomic_bool, 110
 - ATOMIC_BOOL_LOCK_FREE, 110
 - atomic_char, 110
 - atomic_char16_t, 111
 - atomic_char32_t, 111
 - atomic_int, 111
 - atomic_int16_t, 111
 - atomic_int32_t, 111
 - atomic_int64_t, 111
 - atomic_int8_t, 111
 - atomic_int_fast16_t, 112
 - atomic_int_fast32_t, 112
 - atomic_int_fast64_t, 112
 - atomic_int_fast8_t, 112
 - atomic_int_least16_t, 112
 - atomic_int_least32_t, 112
 - atomic_int_least64_t, 112
 - atomic_int_least8_t, 113
 - atomic_intmax_t, 113
 - atomic_intptr_t, 113
 - atomic_llong, 113
 - atomic_long, 113
 - atomic_ptrdiff_t, 113
 - atomic_schar, 113
 - atomic_short, 114
 - atomic_size_t, 114
 - atomic_uchar, 114
 - atomic_uint, 114
 - atomic_uint16_t, 114
 - atomic_uint32_t, 114
 - atomic_uint64_t, 114
 - atomic_uint8_t, 115
 - atomic_uint_fast16_t, 115
 - atomic_uint_fast32_t, 115
 - atomic_uint_fast64_t, 115
 - atomic_uint_fast8_t, 115
 - atomic_uint_least16_t, 115
 - atomic_uint_least32_t, 115
 - atomic_uint_least64_t, 116
 - atomic_uint_least8_t, 116
 - atomic_uintmax_t, 116
 - atomic_uintptr_t, 116
 - atomic_ullong, 116
 - atomic_ulong, 116
 - atomic_ushort, 116
 - atomic_wchar_t, 117
 - kill_dependency, 117
 - memory_order, 117
- auto_ptr
 - std::auto_ptr< _Tp >, 1152, 1153
- auto_ptr.h, 3164
- awk
 - std::regex_constants, 780
- b
 - std::extreme_value_distribution< _RealType >, 2058
 - std::weibull_distribution< _RealType >, 3143
- back
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 895
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1568
 - std::basic_string< _CharT, _Traits, _Alloc >, 1609
 - std::deque< _Tp, _Alloc >, 1976
 - std::list< _Tp, _Alloc >, 2292
 - std::queue< _Tp, _Sequence >, 2652
 - std::vector< _Tp, _Alloc >, 3110
- back_insert_iterator
 - std::back_insert_iterator< _Container >, 1157
- back_inserter
 - Iterators, 177
- backward_warning.h, 3165
- bad
 - std::basic_fstream< _CharT, _Traits >, 1201
 - std::basic_ifstream< _CharT, _Traits >, 1251
 - std::basic_ios< _CharT, _Traits >, 1291
 - std::basic_iostream< _CharT, _Traits >, 1315
 - std::basic_istream< _CharT, _Traits >, 1360
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1398
 - std::basic_ofstream< _CharT, _Traits >, 1440
 - std::basic_ostream< _CharT, _Traits >, 1472
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1505
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1688
- badbit
 - std::basic_fstream< _CharT, _Traits >, 1236
 - std::basic_ifstream< _CharT, _Traits >, 1279

- std::basic_ios< _CharT, _Traits >, 1302
- std::basic_iostream< _CharT, _Traits >, 1348
- std::basic_istream< _CharT, _Traits >, 1386
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1427
- std::basic_ofstream< _CharT, _Traits >, 1461
- std::basic_ostream< _CharT, _Traits >, 1493
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1525
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1721
- std::ios_base, 2198
- balanced_quicksort.h, 3508
- barrier, 3167
- base
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1036
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1049
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1994
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2174
 - std::reverse_iterator< _Iterator >, 2710
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, 2787
- Base and Implementation Classes, 301
 - _Opcode, 301
- Base and Policy Classes, 147–149
- base.h, 3508
- basefield
 - std::basic_fstream< _CharT, _Traits >, 1236
 - std::basic_ifstream< _CharT, _Traits >, 1279
 - std::basic_ios< _CharT, _Traits >, 1302
 - std::basic_iostream< _CharT, _Traits >, 1348
 - std::basic_istream< _CharT, _Traits >, 1386
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1427
 - std::basic_ofstream< _CharT, _Traits >, 1461
 - std::basic_ostream< _CharT, _Traits >, 1493
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1525
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1721
 - std::ios_base, 2198
- basic
 - std::regex_constants, 780
- basic_file.h, 3583
- basic_filebuf
 - std::basic_filebuf< _CharT, _Traits >, 1173
- basic_fstream
 - std::basic_fstream< _CharT, _Traits >, 1199, 1200
- basic_ifstream
 - std::basic_ifstream< _CharT, _Traits >, 1250, 1251
- basic_ios
 - std::basic_ios< _CharT, _Traits >, 1290
- basic_ios.h, 3188
- basic_ios.tcc, 3189
- basic_iostream
 - std::basic_iostream< _CharT, _Traits >, 1314
- basic_istream
 - std::basic_istream< _CharT, _Traits >, 1359
- basic_istreamstream
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1397
- basic_iterator.h, 3509
- basic_ofstream
 - std::basic_ofstream< _CharT, _Traits >, 1439
- basic_ostream
 - std::basic_ostream< _CharT, _Traits >, 1472
- basic_ostreamstream
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1504
- basic_regex
 - std::basic_regex< _Ch_type, _Rx_traits >, 1531–1533
- basic_streambuf
 - std::basic_streambuf< _CharT, _Traits >, 1543
- basic_string
 - std::basic_string< _CharT, _Traits, _Alloc >, 1596, 1597, 1599, 1600
- basic_string.h, 3189
- basic_string.tcc, 3192
- basic_stringbuf
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1663
- basic_stringstream
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1687
- before
 - std::type_info, 2951
- before_begin
 - std::forward_list< _Tp, _Alloc >, 2080
- beg
 - std::basic_fstream< _CharT, _Traits >, 1236
 - std::basic_ifstream< _CharT, _Traits >, 1279
 - std::basic_ios< _CharT, _Traits >, 1303
 - std::basic_iostream< _CharT, _Traits >, 1348
 - std::basic_istream< _CharT, _Traits >, 1386
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1427
 - std::basic_ofstream< _CharT, _Traits >, 1461
 - std::basic_ostream< _CharT, _Traits >, 1493
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1525
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1721
 - std::ios_base, 2198
- begin

- `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >`, 895
- `__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >`, 2869
- `__gnu_parallel::PseudoSequence< _Tp, _DifferenceTp >`, 1019
- `__gnu_pbds::sample_trie_access_traits`, 2729
- `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >`, 2936
- File System, 168
- Numeric Arrays, 231
- std, 646
- `std::Temporary_buffer< _ForwardIterator, _Tp >`, 1083
- `std::basic_string< _CharT, _Traits, _Alloc >`, 1610
- `std::deque< _Tp, _Alloc >`, 1976
- `std::forward_list< _Tp, _Alloc >`, 2080
- `std::initializer_list< _E >`, 2179
- `std::list< _Tp, _Alloc >`, 2292
- `std::map< _Key, _Tp, _Compare, _Alloc >`, 2343
- `std::match_results< _Bi_iter, _Alloc >`, 2371
- `std::multimap< _Key, _Tp, _Compare, _Alloc >`, 2437
- `std::multiset< _Key, _Compare, _Alloc >`, 2464
- `std::set< _Key, _Compare, _Alloc >`, 2751
- `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >`, 2988
- `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3019, 3020
- `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`, 3046, 3047
- `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`, 3074
- `std::vector< _Tp, _Alloc >`, 3111
- Bernoulli Distributions, 243
 - `operator!=`, 244
 - `operator<<`, 244, 245
 - `operator>>`, 245, 246
- `bernoulli_distribution`
 - `std::bernoulli_distribution`, 1726
- beta
 - Mathematical Special Functions, 204
 - `std::gamma_distribution< _RealType >`, 2110
 - TR1 Mathematical Special Functions, 263
- betaf
 - Mathematical Special Functions, 205
- betal
 - Mathematical Special Functions, 205
- `bin_search_tree.hpp`, 3403
- binary
 - `std::basic_fstream< _CharT, _Traits >`, 1236
 - `std::basic_ifstream< _CharT, _Traits >`, 1279
 - `std::basic_ios< _CharT, _Traits >`, 1303
 - `std::basic_iostream< _CharT, _Traits >`, 1349
 - `std::basic_istream< _CharT, _Traits >`, 1386
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1427
 - `std::basic_ofstream< _CharT, _Traits >`, 1461
 - `std::basic_ostream< _CharT, _Traits >`, 1493
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, 1525
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1722
 - `std::ios_base`, 2198
- Binary Search, 86
 - `binary_search`, 87
 - `equal_range`, 88
 - `lower_bound`, 89
 - `upper_bound`, 90
- `binary_heap.hpp`, 3407
- `binary_heap_const_iterator_`
 - `__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >`, 1746
- `binary_heap_point_const_iterator_`
 - `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >`, 1749
- `binary_search`
 - Binary Search, 87
- bind
 - Binder Classes, 357
- `bind1st`
 - Binder Classes, 357
- `bind2nd`
 - Binder Classes, 357
- `bind_front`
 - std, 646
- Binder Classes, 355
 - `bind`, 357
 - `bind1st`, 357
 - `bind2nd`, 357
- `binders.h`, 3165
- `binomial_heap.hpp`, 3420
- `binomial_heap_base.hpp`, 3421
- bit, 3167
- Bit manipulation, 183
 - `bit_cast`, 184
 - `bit_ceil`, 184
 - `bit_floor`, 184
 - `bit_width`, 185
 - `countl_one`, 185
 - `countl_zero`, 185
 - `countr_one`, 185
 - `countr_zero`, 185
 - endian, 184
 - `has_single_bit`, 185
 - popcount, 185
 - `rotr`, 185

- rotr, [186](#)
- bit_cast
 - Bit manipulation, [184](#)
- bit_ceil
 - Bit manipulation, [184](#)
- bit_floor
 - Bit manipulation, [184](#)
- bit_width
 - Bit manipulation, [185](#)
- bitmap_allocator.h, [3399](#)
 - _BALLOC_ALIGN_BYTES, [3399](#)
- bitset, [3340](#), [3341](#)
 - std::bitset< _Nb >, [1768](#), [1769](#)
- bool_constant
 - Metaprogramming, [397](#)
- bool_set, [3552](#)
 - std::tr2::bool_set, [1776](#)
- bool_set.tcc, [3553](#)
- boolalpha
 - std, [647](#)
 - std::basic_fstream< _CharT, _Traits >, [1236](#)
 - std::basic_ifstream< _CharT, _Traits >, [1280](#)
 - std::basic_ios< _CharT, _Traits >, [1303](#)
 - std::basic_iostream< _CharT, _Traits >, [1349](#)
 - std::basic_istream< _CharT, _Traits >, [1386](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1428](#)
 - std::basic_ofstream< _CharT, _Traits >, [1461](#)
 - std::basic_ostream< _CharT, _Traits >, [1493](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1526](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1722](#)
 - std::ios_base, [2198](#)
- Boolean Operations Classes, [357](#)
- boost_concept_check.h, [3193](#)
- Branch-Based, [146](#)
- branch_policy.hpp, [3421](#)
- bucket
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, [1049](#)
- bucket_count
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2990](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3020](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3047](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3074](#)
- byte
 - std, [634](#)
- c
 - std::queue< _Tp, _Sequence >, [2653](#)
 - c++0x_warning.h, [3193](#)
 - c++allocator.h, [3583](#)
 - c++config.h, [3583](#)
 - c++io.h, [3589](#)
 - c++locale.h, [3590](#)
 - c++locale_internal.h, [3590](#)
 - c_str
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [895](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1610](#)
 - cache_line_size
 - __gnu_parallel::Settings, [1075](#)
 - call_once
 - Mutexes, [123](#)
 - std::once_flag, [2564](#)
 - capacity
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [895](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1569](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1610](#)
 - std::vector< _Tp, _Alloc >, [3111](#)
 - cassert, [3341](#)
 - cast.h, [3399](#)
 - category
 - std::error_code, [2048](#)
 - std::error_condition, [2050](#)
 - std::locale, [2309](#)
 - cbefore_begin
 - std::forward_list< _Tp, _Alloc >, [2080](#)
 - cbegin
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [896](#)
 - std, [647](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1610](#)
 - std::deque< _Tp, _Alloc >, [1977](#)
 - std::forward_list< _Tp, _Alloc >, [2081](#)
 - std::list< _Tp, _Alloc >, [2292](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2343](#)
 - std::match_results< _Bi_iter, _Alloc >, [2371](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2437](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2465](#)
 - std::set< _Key, _Compare, _Alloc >, [2751](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2990](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3020](#), [3021](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3047](#), [3048](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3075](#)
 - std::vector< _Tp, _Alloc >, [3111](#)

- cc_hash_max_collision_check_resize_trigger
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger<std::money_get< _CharT, _InIter >, 2398
External_Load_Access, Size_Type >, 1782
- cc_hash_max_collision_check_resize_trigger_imp.hpp, 3437
- cc_hash_table
 - __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Re-size_Policy, Store_Hash, _Alloc >, 1786–1788
- cc_ht_map.hpp, 3422
- ccomplex, 3341, 3342
- cctype, 3342
- ceil
 - Time, 414, 415
- cend
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 896
 - std, 647
 - std::basic_string< _CharT, _Traits, _Alloc >, 1610
 - std::deque< _Tp, _Alloc >, 1977
 - std::forward_list< _Tp, _Alloc >, 2081
 - std::list< _Tp, _Alloc >, 2293
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2344
 - std::match_results< _Bi_iter, _Alloc >, 2371
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2437
 - std::multiset< _Key, _Compare, _Alloc >, 2465
 - std::set< _Key, _Compare, _Alloc >, 2751
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2990
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3021
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3048
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3075
 - std::vector< _Tp, _Alloc >, 3111
- cerr
 - std, 710
- cerrno, 3342
- cfenv, 3342, 3343
- cfloat, 3343
- char_traits.h, 3193
- char_type
 - std::__ctype_abstract_base< _CharT >, 826
 - std::basic_ios< _CharT, _Traits >, 1288
 - std::basic_streambuf< _CharT, _Traits >, 1542
 - std::collate< _CharT >, 1839
 - std::collate_byname< _CharT >, 1845
 - std::ctype< char >, 1900
 - std::ctype< wchar_t >, 1912
 - std::ctype_byname< char >, 1939
 - std::istreambuf_iterator< _CharT, _Traits >, 2254
 - std::messages< _CharT >, 2388
 - std::money_get< _CharT, _InIter >, 2398
 - std::money_put< _CharT, _OutIter >, 2402
 - std::moneypunct< _CharT, _Intl >, 2407
 - std::num_get< _CharT, _InIter >, 2502
 - std::num_put< _CharT, _OutIter >, 2517
 - std::numpunct< _CharT >, 2553
 - std::ostream_iterator< _Tp, _CharT, _Traits >, 2569
 - std::ostreambuf_iterator< _CharT, _Traits >, 2573
 - std::time_get< _CharT, _InIter >, 2883
 - std::time_put< _CharT, _OutIter >, 2905
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3127
- charconv, 3343
- charconv.h, 3194
- chars_format
 - std, 634
- checkers.h, 3509
- chrono, 3345, 3346
- chrono.h, 3194
- cin
 - std, 710
- cinttypes, 3347
- ciso646, 3347
- clamp
 - Sorting, 64, 65
- classic
 - std::locale, 2312
- classic_table
 - std::ctype< char >, 1901
 - std::ctype_byname< char >, 1939
- clear
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 896
 - std::basic_fstream< _CharT, _Traits >, 1201
 - std::basic_ifstream< _CharT, _Traits >, 1251
 - std::basic_ios< _CharT, _Traits >, 1291
 - std::basic_iostream< _CharT, _Traits >, 1315
 - std::basic_istream< _CharT, _Traits >, 1360
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1398
 - std::basic_ofstream< _CharT, _Traits >, 1440
 - std::basic_ostream< _CharT, _Traits >, 1472
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1505
 - std::basic_string< _CharT, _Traits, _Alloc >, 1610
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1688
 - std::deque< _Tp, _Alloc >, 1977
 - std::error_condition, 2050
 - std::experimental::fundamentals_v1::any, 1116
 - std::forward_list< _Tp, _Alloc >, 2081
 - std::list< _Tp, _Alloc >, 2293
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2344
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2437

- std::multiset< _Key, _Compare, _Alloc >, 2465
- std::set< _Key, _Compare, _Alloc >, 2751
- std::tr2::dynamic_bitset< _WordT, _Alloc >, 2009
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2991
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3021
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3048
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3075
- std::vector< _Tp, _Alloc >, 3111
- climits, 3347, 3348
- locale, 3348
- clog
 - std, 710
- close
 - __gnu_cxx::enc_filebuf< _CharT >, 2024
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2808
 - std::basic_filebuf< _CharT, _Traits >, 1174
 - std::basic_fstream< _CharT, _Traits >, 1201
 - std::basic_ifstream< _CharT, _Traits >, 1252
 - std::basic_ofstream< _CharT, _Traits >, 1440
- cmath, 3348, 3350, 3351
- cmp_fn
 - __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >, 2914
- cmp_fn_imps.hpp, 3422
- code
 - std::regex_error, 2691
- codecvt, 3353
- codecvt.h, 3198
- codecvt_specializations.h, 3400
- collate
 - std::collate< _CharT >, 1839, 1840
 - std::locale, 2315
 - std::regex_constants, 781
- combine
 - std::locale, 2312
- common_type_t
 - Metaprogramming, 397
- comp_ellint_1
 - Mathematical Special Functions, 205
 - TR1 Mathematical Special Functions, 264
- comp_ellint_1f
 - Mathematical Special Functions, 206
- comp_ellint_1l
 - Mathematical Special Functions, 206
- comp_ellint_2
 - Mathematical Special Functions, 206
 - TR1 Mathematical Special Functions, 264
- comp_ellint_2f
 - Mathematical Special Functions, 207
- comp_ellint_2l
 - Mathematical Special Functions, 207
- comp_ellint_3
 - Mathematical Special Functions, 207
 - TR1 Mathematical Special Functions, 264
- comp_ellint_3f
 - Mathematical Special Functions, 208
- comp_ellint_3l
 - Mathematical Special Functions, 208
- compare, 3148
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 896–899
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1569–1571
 - std::basic_string< _CharT, _Traits, _Alloc >, 1610–1614
 - std::collate< _CharT >, 1840
 - std::collate_byname< _CharT >, 1846
 - std::sub_match< _Bilter >, 2853, 2854
- compare_three_way_result_t
 - std, 633
- Comparison Classes, 358
- compatibility.h, 3590, 3591
- compiletime_settings.h, 3509
 - _GLIBCXX_CALL, 3510
 - _GLIBCXX_PARALLEL_ASSERTIONS, 3510
 - _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1, 3510
 - _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB, 3510
 - _GLIBCXX_SCALE_DOWN_FPU, 3510
 - _GLIBCXX_VERBOSE_LEVEL, 3510
- complex, 3353, 3357
 - std::complex< _Tp >, 1853
- Complex Numbers, 186
 - abs, 189
 - arg, 189
 - conj, 189
 - cos, 190
 - cosh, 190
 - exp, 190
 - fabs, 190
 - log, 190
 - log10, 190
 - norm, 190
 - operator<=, 193
 - operator>=, 194
 - operator*, 191
 - operator*=, 191
 - operator+, 191, 192
 - operator+=, 192
 - operator-, 192
 - operator-=, 193
 - operator/, 193
 - operator/=, 193

- operator=, [194](#)
- operator==, [194](#)
- polar, [194](#)
- pow, [194](#), [195](#)
- sin, [195](#)
- sinh, [195](#)
- sqrt, [195](#)
- tan, [196](#)
- tanh, [196](#)
- complex.h, [3358](#)
- compose1
 - SGI, [159](#)
- compose2
 - SGI, [159](#)
- concept_check.h, [3198](#)
- Concepts, [350](#)
- concurrency.h, [3400](#)
- Concurrency, [117](#)
- cond_dealtor.hpp, [3425](#)
- cond_key_dtor_entry_dealtor.hpp, [3422](#)
- Condition Variables, [118](#)
 - cv_status, [119](#)
- condition_variable, [3360](#)
- conditional_t
 - Metaprogramming, [398](#)
- conf_hyperg
 - Mathematical Special Functions, [208](#)
 - TR1 Mathematical Special Functions, [264](#)
- conf_hypergfl
 - Mathematical Special Functions, [208](#)
- conf_hypergll
 - Mathematical Special Functions, [209](#)
- conj
 - Complex Numbers, [189](#)
- Const-propagating wrapper, [311](#)
- const_iterator
 - __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >, [2936](#)
 - std::set< _Key, _Compare, _Alloc >, [2746](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2983](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3015](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3042](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- const_iterator.hpp, [3407](#), [3408](#)
- const_local_iterator
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2983](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3015](#)
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- const_pointer
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1745](#)
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1748](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, [2265](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, [2269](#)
 - std::allocator_traits< _Alloc >, [1094](#)
 - std::allocator_traits< allocator< _Tp > >, [1099](#)
 - std::allocator_traits< allocator< void > >, [1103](#)
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, [1107](#)
 - std::set< _Key, _Compare, _Alloc >, [2746](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2983](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3015](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- const_pointer_cast
 - Pointer Abstractions, [376](#)
- std, [647](#)
- const_reference
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1731](#)
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1735](#)
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1745](#)
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1748](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, [2265](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, [2269](#)
 - std::set< _Key, _Compare, _Alloc >, [2746](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2983](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- const_reverse_iterator
 - std::set< _Key, _Compare, _Alloc >, [2747](#)

- const_void_pointer
 - `__gnu_cxx::__alloc_traits<_Alloc, typename>`, 806
 - `std::allocator_traits<_Alloc>`, 1094
 - `std::allocator_traits<allocator<_Tp>>`, 1099
 - `std::allocator_traits<allocator<void>>`, 1103
 - `std::allocator_traits<pmr::polymorphic_allocator<_Tp>>`, 1107
- constant0
 - SGL, 160
- constant1
 - SGL, 160
- constant2
 - SGL, 160
- construct
 - `__gnu_cxx::__alloc_traits<_Alloc, typename>`, 808
 - `std::allocator_traits<_Alloc>`, 1096
 - `std::allocator_traits<allocator<_Tp>>`, 1101
 - `std::allocator_traits<allocator<void>>`, 1104
 - `std::allocator_traits<pmr::polymorphic_allocator<_Tp>>`, 1109
- constructor_destructor_fn_imps.hpp, 3422, 3423
- constructor_destructor_no_store_hash_fn_imps.hpp, 3423
- constructor_destructor_store_hash_fn_imps.hpp, 3423
- constructors_destructor_fn_imps.hpp, 3408, 3409
- container_base_dispatch.hpp, 3426
- container_type
 - `std::back_insert_iterator<_Container>`, 1157
 - `std::front_insert_iterator<_Container>`, 2096
 - `std::insert_iterator<_Container>`, 2183
- Containers, 126, 146
- contains
 - `std::map<_Key, _Tp, _Compare, _Alloc>`, 2344
 - `std::multimap<_Key, _Tp, _Compare, _Alloc>`, 2437, 2438
 - `std::multiset<_Key, _Compare, _Alloc>`, 2465
 - `std::set<_Key, _Compare, _Alloc>`, 2751, 2752
 - `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>`, 2991
 - `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>`, 3022
 - `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>`, 3049
 - `std::unordered_set<_Value, _Hash, _Pred, _Alloc>`, 3076
- converted
 - `std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>`, 3147
- copy
 - `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>`, 899
 - Mutating, 22
 - `std::basic_string<_CharT, _Traits, _Alloc>`, 1615
- copy_backward
 - Mutating, 23
- copy_if
 - Mutating, 23
- copy_n
 - Mutating, 24
 - SGL, 160
- copy_options
 - File System, 168
 - Filesystem TS, 308
- copyfmt
 - `std::basic_fstream<_CharT, _Traits>`, 1201
 - `std::basic_ifstream<_CharT, _Traits>`, 1252
 - `std::basic_ios<_CharT, _Traits>`, 1291
 - `std::basic_iostream<_CharT, _Traits>`, 1315
 - `std::basic_istream<_CharT, _Traits>`, 1360
 - `std::basic_istreamstream<_CharT, _Traits, _Alloc>`, 1399
 - `std::basic_ofstream<_CharT, _Traits>`, 1441
 - `std::basic_ostream<_CharT, _Traits>`, 1473
 - `std::basic_ostreamstream<_CharT, _Traits, _Alloc>`, 1505
 - `std::basic_stringstream<_CharT, _Traits, _Alloc>`, 1688
- cos
 - Complex Numbers, 190
- cosh
 - Complex Numbers, 190
- count
 - Non-Mutating, 46
 - `std::bitset<_Nb>`, 1770
 - `std::map<_Key, _Tp, _Compare, _Alloc>`, 2345
 - `std::multimap<_Key, _Tp, _Compare, _Alloc>`, 2438
 - `std::multiset<_Key, _Compare, _Alloc>`, 2466
 - `std::set<_Key, _Compare, _Alloc>`, 2752
 - `std::tr2::dynamic_bitset<_WordT, _Alloc>`, 2009
 - `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>`, 2992
 - `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>`, 3022, 3023
 - `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>`, 3049, 3050
 - `std::unordered_set<_Value, _Hash, _Pred, _Alloc>`, 3076, 3077
- count_if
 - Non-Mutating, 47
- count_minimal_n
 - `__gnu_parallel::Settings`, 1075
- countl_one
 - Bit manipulation, 185
- countl_zero
 - Bit manipulation, 185
- countr_one
 - Bit manipulation, 185

- countr_zero
 - Bit manipulation, [185](#)
- cout
 - std, [710](#)
- cow_string.h, [3198](#)
- cpp_type_traits.h, [3199](#)
- cpu_defines.h, [3591](#)
- crbegin
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [900](#)
 - std, [647](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1615](#)
 - std::deque< _Tp, _Alloc >, [1977](#)
 - std::list< _Tp, _Alloc >, [2293](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2345](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2439](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2466](#)
 - std::set< _Key, _Compare, _Alloc >, [2753](#)
 - std::vector< _Tp, _Alloc >, [3111](#)
- cref
 - std::reference_wrapper< _Tp >, [2689](#)
- cregex_token_iterator
 - Regular Expressions, [272](#)
- crend
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [900](#)
 - std, [648](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1615](#)
 - std::deque< _Tp, _Alloc >, [1977](#)
 - std::list< _Tp, _Alloc >, [2293](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2345](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2439](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2466](#)
 - std::set< _Key, _Compare, _Alloc >, [2753](#)
 - std::vector< _Tp, _Alloc >, [3112](#)
- csetjmp, [3361](#)
- cshift
 - Numeric Arrays, [231](#)
- csignal, [3361](#)
- cstdalign, [3361](#)
- cstdarg, [3362](#)
- cstdbool, [3362](#)
- cstddef, [3362](#)
- cstdint, [3363](#)
- cstdio, [3364](#)
- cstdlib, [3364](#), [3365](#)
- cstring, [3365](#)
- csub_match
 - Regular Expressions, [272](#)
- ctgmath, [3365](#)
- ctime, [3366](#)
- ctype
 - std::ctype< char >, [1900](#)
 - std::ctype< wchar_t >, [1912](#)
 - std::locale, [2316](#)
- ctype_base.h, [3591](#)
- ctype_inline.h, [3591](#)
- cuchar, [3366](#)
- cur
 - std::basic_fstream< _CharT, _Traits >, [1236](#)
 - std::basic_ifstream< _CharT, _Traits >, [1280](#)
 - std::basic_ios< _CharT, _Traits >, [1303](#)
 - std::basic_iostream< _CharT, _Traits >, [1349](#)
 - std::basic_istream< _CharT, _Traits >, [1386](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1428](#)
 - std::basic_ofstream< _CharT, _Traits >, [1462](#)
 - std::basic_ostream< _CharT, _Traits >, [1494](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1526](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1722](#)
 - std::ios_base, [2198](#)
- curr_symbol
 - std::moneypunct< _CharT, _Intl >, [2409](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2416](#)
- current_exception
 - Exceptions, [137](#)
- cv_status
 - Condition Variables, [119](#)
- cwchar, [3366](#), [3367](#)
- cwctype, [3367](#)
- cxxabi.h, [3150](#)
 - __cxa_demangle, [3151](#)
- cxxabi_forced.h, [3199](#)
- cxxabi_init_exception.h, [3199](#)
- cxxabi_tweaks.h, [3592](#)
- cyl_bessel_i
 - Mathematical Special Functions, [209](#)
 - TR1 Mathematical Special Functions, [264](#)
- cyl_bessel_if
 - Mathematical Special Functions, [210](#)
- cyl_bessel_il
 - Mathematical Special Functions, [210](#)
- cyl_bessel_j
 - Mathematical Special Functions, [210](#)
 - TR1 Mathematical Special Functions, [264](#)
- cyl_bessel_jf
 - Mathematical Special Functions, [210](#)
- cyl_bessel_jl
 - Mathematical Special Functions, [211](#)
- cyl_bessel_k
 - Mathematical Special Functions, [211](#)
 - TR1 Mathematical Special Functions, [264](#)
- cyl_bessel_kf
 - Mathematical Special Functions, [211](#)

- cyl_bessel_kl
 - Mathematical Special Functions, [212](#)
- cyl_neumann
 - Mathematical Special Functions, [212](#)
 - TR1 Mathematical Special Functions, [265](#)
- cyl_neumannf
 - Mathematical Special Functions, [212](#)
- cyl_neumannl
 - Mathematical Special Functions, [213](#)
- data
 - `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >`, [900](#)
 - `__gnu_debug::basic_string< _CharT, _Traits, _Allocator >`, [1571](#)
 - std, [648](#)
 - `std::basic_string< _CharT, _Traits, _Alloc >`, [1616](#)
 - `std::vector< _Tp, _Alloc >`, [3112](#)
- Data parallel extensions, [323](#)
- Data Structure Type, [152](#)
- date_order
 - `std::time_get< _CharT, _Intl >`, [2884](#)
 - `std::time_get_byname< _CharT, _Intl >`, [2894](#)
- days
 - Time, [413](#)
- deallocate
 - `__gnu_cxx::__alloc_traits< _Alloc, typename >`, [808](#), [809](#)
 - `std::allocator_traits< _Alloc >`, [1096](#)
 - `std::allocator_traits< allocator< _Tp > >`, [1101](#)
 - `std::allocator_traits< allocator< void > >`, [1105](#)
 - `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, [1109](#)
- debug.h, [3368](#)
- debug_allocator.h, [3401](#)
- debug_fn_imps.hpp, [3409–3411](#)
- debug_map_base.hpp, [3426](#)
- debug_no_store_hash_fn_imps.hpp, [3423](#)
- debug_store_hash_fn_imps.hpp, [3423](#)
- dec
 - std, [649](#)
 - `std::basic_fstream< _CharT, _Traits >`, [1237](#)
 - `std::basic_ifstream< _CharT, _Traits >`, [1280](#)
 - `std::basic_ios< _CharT, _Traits >`, [1303](#)
 - `std::basic_iostream< _CharT, _Traits >`, [1349](#)
 - `std::basic_istream< _CharT, _Traits >`, [1386](#)
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, [1428](#)
 - `std::basic_ofstream< _CharT, _Traits >`, [1462](#)
 - `std::basic_ostream< _CharT, _Traits >`, [1494](#)
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, [1526](#)
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, [1722](#)
 - `std::ios_base`, [2199](#)
- decay_t
 - Metaprogramming, [398](#)
- decimal, [3385](#)
- Decimal Floating-Point Arithmetic, [196](#)
- decimal128
 - `std::decimal::decimal128`, [1951](#)
- decimal32
 - `std::decimal::decimal32`, [1952](#)
- decimal32_to_long_long
 - `std::decimal`, [754](#)
- decimal64
 - `std::decimal::decimal64`, [1954](#)
- decimal_point
 - `std::moneypunct< _CharT, _Intl >`, [2409](#)
 - `std::moneypunct_byname< _CharT, _Intl >`, [2416](#)
 - `std::numpunct< _CharT >`, [2554](#)
 - `std::numpunct_byname< _CharT >`, [2558](#)
- declare_no_pointers
 - Pointer Safety and Garbage Collection, [388](#)
- declare_reachable
 - Pointer Safety and Garbage Collection, [388](#)
- declval
 - Utilities, [344](#)
- default_delete
 - `std::default_delete< _Tp >`, [1955](#)
 - `std::default_delete< _Tp[] >`, [1956](#)
- default_error_condition
 - `std::_V2::error_category`, [2047](#)
 - `std::error_code`, [2048](#)
- default_sentinel
 - std, [711](#)
- defaultfloat
 - std, [649](#)
- defer_lock
 - Mutexes, [125](#)
- denorm_absent
 - std, [634](#)
- denorm_indeterminate
 - std, [634](#)
- denorm_min
 - `std::numeric_limits< _Tp >`, [2529](#)
- denorm_present
 - std, [634](#)
- densities
 - `std::piecewise_constant_distribution< _RealType >`, [2621](#)
 - `std::piecewise_linear_distribution< _RealType >`, [2625](#)
- Deprecated List, [1](#)
- deque, [3394](#), [3395](#)
 - `std::deque< _Tp, _Alloc >`, [1968–1970](#)
- deque.tcc, [3200](#)
- destroy

- `__gnu_cxx::__alloc_traits< _Alloc, typename >`, 809
- `std::allocator_traits< _Alloc >`, 1097
- `std::allocator_traits< allocator< _Tp > >`, 1101
- `std::allocator_traits< allocator< void > >`, 1105
- `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, 1109
- destroying_delete
 - std, 711
- detected_or
 - Detection idiom, 313
- detected_or_t
 - Detection idiom, 313
- detected_t
 - Detection idiom, 314
- Detection idiom, 313
 - `__cpp_lib_experimental_detect`, 313
 - detected_or, 313
 - detected_or_t, 313
 - detected_t, 314
 - is_detected, 314
 - is_detected_convertible, 314
 - is_detected_convertible_v, 314
 - is_detected_exact, 314
 - is_detected_exact_v, 314
 - is_detected_v, 314
 - void_t, 314
- Diagnostics, 130
 - generic_category, 131
 - make_error_code, 131
 - make_error_condition, 132
 - operator<, 132
 - operator<=, 132
 - operator==, 133
 - system_category, 134
- difference_type
 - `__gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >`, 1731
 - `__gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >`, 1735
 - `__gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >`, 1745
 - `__gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >`, 1748
 - `__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >`, 2265
 - `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >`, 2269
 - `std::allocator_traits< _Alloc >`, 1094
 - `std::allocator_traits< allocator< _Tp > >`, 1099
 - `std::allocator_traits< allocator< void > >`, 1103
 - `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, 1107
 - `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >`, 2251
 - `std::istreambuf_iterator< _CharT, _Traits >`, 2254
 - `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`, 2259
 - `std::ostream_iterator< _Tp, _CharT, _Traits >`, 2569
 - `std::ostreambuf_iterator< _CharT, _Traits >`, 2573
 - `std::pointer_traits< _Ptr >`, 2632
 - `std::pointer_traits< _Tp * >`, 2633
 - `std::raw_storage_iterator< _OutputIterator, _Tp >`, 2670
 - `std::set< _Key, _Compare, _Alloc >`, 2747
 - `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >`, 2983
 - `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3016
 - `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`, 3043
 - `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`, 3070
- digits
 - `std::__numeric_limits_base`, 863
 - `std::numeric_limits< _Tp >`, 2531
- digits10
 - `std::__numeric_limits_base`, 863
 - `std::numeric_limits< _Tp >`, 2531
- `direct_mask_range_hashing_imp.hpp`, 3428
- `direct_mod_range_hashing_imp.hpp`, 3428
- directory_options
 - File System, 168
 - Filesystem TS, 308
- discard
 - `std::discard_block_engine< _RandomNumberEngine, __p, __r >`, 1994
 - `std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >`, 2174
 - `std::linear_congruential_engine< _UIntType, __a, __c, __m >`, 2278
 - `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >`, 2384
 - `std::shuffle_order_engine< _RandomNumberEngine, __k >`, 2787
 - `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >`, 2860
 - `std::discard_block_engine< _RandomNumberEngine, __p, __r >`, 1994
- distance
 - SGI, 160
 - std, 649
- do_allocate
 - `std::pmr::synchronized_pool_resource`, 2865
 - `std::pmr::unsynchronized_pool_resource`, 3091
- do_always_noconv
 - `std::codecvt< _InternT, _ExternT, _StateT >`, 1804

- std::codecvt< _InternT, _ExternT, encoding_state >, 1809
- std::codecvt< char, char, mbstate_t >, 1814
- std::codecvt< char16_t, char, mbstate_t >, 1818
- std::codecvt< char32_t, char, mbstate_t >, 1823
- std::codecvt< wchar_t, char, mbstate_t >, 1828
- std::codecvt_byname< _InternT, _ExternT, _StateT >, 1834
- do_compare
 - std::collate< _CharT >, 1840
 - std::collate_byname< _CharT >, 1846
- do_curr_symbol
 - std::moneypunct< _CharT, _Intl >, 2409
 - std::moneypunct_byname< _CharT, _Intl >, 2417
- do_date_order
 - std::time_get< _CharT, _InIter >, 2884
 - std::time_get_byname< _CharT, _InIter >, 2895
- do_deallocate
 - std::pmr::synchronized_pool_resource, 2865
 - std::pmr::unsynchronized_pool_resource, 3091
- do_decimal_point
 - std::moneypunct< _CharT, _Intl >, 2409
 - std::moneypunct_byname< _CharT, _Intl >, 2417
 - std::numpunct< _CharT >, 2554
 - std::numpunct_byname< _CharT >, 2559
- do_encoding
 - std::codecvt< _InternT, _ExternT, _StateT >, 1804
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1809
 - std::codecvt< char, char, mbstate_t >, 1814
 - std::codecvt< char16_t, char, mbstate_t >, 1818
 - std::codecvt< char32_t, char, mbstate_t >, 1823
 - std::codecvt< wchar_t, char, mbstate_t >, 1828
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, 1834
- do_falsename
 - std::numpunct< _CharT >, 2555
 - std::numpunct_byname< _CharT >, 2559
- do_frac_digits
 - std::moneypunct< _CharT, _Intl >, 2409
 - std::moneypunct_byname< _CharT, _Intl >, 2417
- do_get
 - std::messages< _CharT >, 2389
 - std::messages_byname< _CharT >, 2391
 - std::money_get< _CharT, _InIter >, 2399
 - std::num_get< _CharT, _InIter >, 2502–2508
 - std::time_get< _CharT, _InIter >, 2884
 - std::time_get_byname< _CharT, _InIter >, 2895
- do_get_date
 - std::time_get< _CharT, _InIter >, 2885
 - std::time_get_byname< _CharT, _InIter >, 2895
- do_get_monthname
 - std::time_get< _CharT, _InIter >, 2886
 - std::time_get_byname< _CharT, _InIter >, 2896
- do_get_time
 - std::time_get< _CharT, _InIter >, 2886
 - std::time_get_byname< _CharT, _InIter >, 2897
- do_get_weekday
 - std::time_get< _CharT, _InIter >, 2887
 - std::time_get_byname< _CharT, _InIter >, 2897
- do_get_year
 - std::time_get< _CharT, _InIter >, 2888
 - std::time_get_byname< _CharT, _InIter >, 2898
- do_grouping
 - std::moneypunct< _CharT, _Intl >, 2410
 - std::moneypunct_byname< _CharT, _Intl >, 2418
 - std::numpunct< _CharT >, 2555
 - std::numpunct_byname< _CharT >, 2559
- do_hash
 - std::collate< _CharT >, 1841
 - std::collate_byname< _CharT >, 1846
- do_in
 - std::codecvt< _InternT, _ExternT, _StateT >, 1804
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1809
 - std::codecvt< char, char, mbstate_t >, 1814
 - std::codecvt< char16_t, char, mbstate_t >, 1819
 - std::codecvt< char32_t, char, mbstate_t >, 1823
 - std::codecvt< wchar_t, char, mbstate_t >, 1828
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, 1834
- do_is
 - std::__ctype_abstract_base< _CharT >, 826
 - std::ctype< _CharT >, 1887
 - std::ctype< wchar_t >, 1912, 1913
 - std::ctype_byname< _CharT >, 1926
- do_is_equal
 - std::pmr::synchronized_pool_resource, 2865
 - std::pmr::unsynchronized_pool_resource, 3092
- do_length
 - std::codecvt< _InternT, _ExternT, _StateT >, 1804
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1809
 - std::codecvt< char, char, mbstate_t >, 1814
 - std::codecvt< char16_t, char, mbstate_t >, 1819
 - std::codecvt< char32_t, char, mbstate_t >, 1823
 - std::codecvt< wchar_t, char, mbstate_t >, 1829
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, 1835
- do_max_length
 - std::codecvt< _InternT, _ExternT, _StateT >, 1804
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1809
 - std::codecvt< char, char, mbstate_t >, 1814
 - std::codecvt< char16_t, char, mbstate_t >, 1819
 - std::codecvt< char32_t, char, mbstate_t >, 1824
 - std::codecvt< wchar_t, char, mbstate_t >, 1829

- std::codecvt_byname< _InternT, _ExternT, _StateT >, 1835
- do_narrow
 - std::__ctype_abstract_base< _CharT >, 827
 - std::ctype< _CharT >, 1887, 1888
 - std::ctype< char >, 1901
 - std::ctype< wchar_t >, 1913, 1914
 - std::ctype_byname< _CharT >, 1926, 1927
 - std::ctype_byname< char >, 1939
- do_neg_format
 - std::moneypunct< _CharT, _Intl >, 2410
 - std::moneypunct_byname< _CharT, _Intl >, 2418
- do_negative_sign
 - std::moneypunct< _CharT, _Intl >, 2410
 - std::moneypunct_byname< _CharT, _Intl >, 2418
- do_out
 - std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >, 818
 - std::codecvt< _InternT, _ExternT, _StateT >, 1804
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1809
 - std::codecvt< char, char, mbstate_t >, 1814
 - std::codecvt< char16_t, char, mbstate_t >, 1819
 - std::codecvt< char32_t, char, mbstate_t >, 1824
 - std::codecvt< wchar_t, char, mbstate_t >, 1829
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, 1835
- do_pos_format
 - std::moneypunct< _CharT, _Intl >, 2411
 - std::moneypunct_byname< _CharT, _Intl >, 2418
- do_positive_sign
 - std::moneypunct< _CharT, _Intl >, 2411
 - std::moneypunct_byname< _CharT, _Intl >, 2419
- do_put
 - std::money_put< _CharT, _Outlter >, 2403
 - std::num_put< _CharT, _Outlter >, 2518–2522
 - std::time_put< _CharT, _Outlter >, 2906
 - std::time_put_byname< _CharT, _Outlter >, 2909
- do_scan_is
 - std::__ctype_abstract_base< _CharT >, 828
 - std::ctype< _CharT >, 1888
 - std::ctype< wchar_t >, 1914
 - std::ctype_byname< _CharT >, 1927
- do_scan_not
 - std::__ctype_abstract_base< _CharT >, 828
 - std::ctype< _CharT >, 1889
 - std::ctype< wchar_t >, 1915
 - std::ctype_byname< _CharT >, 1928
- do_thousands_sep
 - std::moneypunct< _CharT, _Intl >, 2411
 - std::moneypunct_byname< _CharT, _Intl >, 2419
 - std::numput< _CharT >, 2555
 - std::numput_byname< _CharT >, 2559
- do_tolower
 - std::__ctype_abstract_base< _CharT >, 829
 - std::ctype< _CharT >, 1889, 1890
 - std::ctype< char >, 1902
 - std::ctype< wchar_t >, 1915, 1916
 - std::ctype_byname< _CharT >, 1928, 1929
 - std::ctype_byname< char >, 1940
- do_toupper
 - std::__ctype_abstract_base< _CharT >, 830
 - std::ctype< _CharT >, 1890, 1891
 - std::ctype< char >, 1903
 - std::ctype< wchar_t >, 1916
 - std::ctype_byname< _CharT >, 1929, 1930
 - std::ctype_byname< char >, 1941
- do_transform
 - std::collate< _CharT >, 1841
 - std::collate_byname< _CharT >, 1847
- do_truename
 - std::numput< _CharT >, 2555
 - std::numput_byname< _CharT >, 2560
- do_unshift
 - std::codecvt< _InternT, _ExternT, _StateT >, 1805
 - std::codecvt< _InternT, _ExternT, encoding_state >, 1810
 - std::codecvt< char, char, mbstate_t >, 1815
 - std::codecvt< char16_t, char, mbstate_t >, 1819
 - std::codecvt< char32_t, char, mbstate_t >, 1824
 - std::codecvt< wchar_t, char, mbstate_t >, 1829
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, 1835
- do_widen
 - std::__ctype_abstract_base< _CharT >, 831
 - std::ctype< _CharT >, 1891, 1892
 - std::ctype< char >, 1903, 1904
 - std::ctype< wchar_t >, 1917
 - std::ctype_byname< _CharT >, 1930, 1931
 - std::ctype_byname< char >, 1941, 1942
- duration_cast
 - Time, 415
- Dynamic Bitset., 140
 - operator!=, 142
 - operator<<, 143
 - operator<=, 143
 - operator>, 143
 - operator>>, 143
 - operator>=, 143
 - operator^, 144
 - operator-, 142
 - operator&, 142
 - operator|, 144
- dynamic_bitset, 3553
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2007, 2008
- dynamic_bitset.tcc, 3554
- dynamic_pointer_cast

- Pointer Abstractions, [376](#)
- std, [649](#)
- e_pos
 - __gnu_pbds::sample_trie_access_traits, [2729](#)
 - __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >, [2936](#)
- e_type
 - __gnu_pbds::sample_trie_access_traits, [2729](#)
 - __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >, [2936](#)
- e_v
 - std::numbers, [772](#)
- eback
 - __gnu_cxx::enc_filebuf< _CharT >, [2024](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2808](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2827](#)
 - std::basic_filebuf< _CharT, _Traits >, [1174](#)
 - std::basic_streambuf< _CharT, _Traits >, [1543](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1664](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3129](#)
- ECMAScript
 - std::regex_constants, [781](#)
- egamma_v
 - std::numbers, [772](#)
- egptr
 - __gnu_cxx::enc_filebuf< _CharT >, [2024](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2809](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2827](#)
 - std::basic_filebuf< _CharT, _Traits >, [1175](#)
 - std::basic_streambuf< _CharT, _Traits >, [1543](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1664](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3129](#)
- egrep
 - std::regex_constants, [781](#)
- element_type
 - std::auto_ptr< _Tp >, [1152](#)
 - std::pointer_traits< _Ptr >, [2632](#)
 - std::pointer_traits< _Tp * >, [2633](#)
 - std::shared_ptr< _Tp >, [2776](#)
- ellint_1
 - Mathematical Special Functions, [213](#)
 - TR1 Mathematical Special Functions, [265](#)
- ellint_1f
 - Mathematical Special Functions, [213](#)
- ellint_1l
 - Mathematical Special Functions, [214](#)
- ellint_2
 - Mathematical Special Functions, [214](#)
 - TR1 Mathematical Special Functions, [265](#)
- ellint_2f
 - Mathematical Special Functions, [215](#)
- ellint_2l
 - Mathematical Special Functions, [215](#)
- ellint_3
 - Mathematical Special Functions, [215](#)
 - TR1 Mathematical Special Functions, [265](#)
- ellint_3f
 - Mathematical Special Functions, [216](#)
- ellint_3l
 - Mathematical Special Functions, [216](#)
- emplace
 - std::any, [1113](#)
 - std::deque< _Tp, _Alloc >, [1977](#)
 - std::list< _Tp, _Alloc >, [2293](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2346](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2439](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2467](#)
 - std::set< _Key, _Compare, _Alloc >, [2753](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2992](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3023](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3050](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3077](#)
 - std::vector< _Tp, _Alloc >, [3112](#)
- emplace_after
 - std::forward_list< _Tp, _Alloc >, [2081](#)
- emplace_front
 - std::forward_list< _Tp, _Alloc >, [2081](#)
- emplace_hint
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2346](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2439](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2467](#)
 - std::set< _Key, _Compare, _Alloc >, [2753](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2993](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3023](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3050](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3077](#)
- empty
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [900](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1571](#)
 - __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash,

- Comb_Hash_Fn, Resize_Policy >, [1792](#)
- __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2123](#)
- std, [650](#)
- std::basic_string< _CharT, _Traits, _Alloc >, [1616](#)
- std::deque< _Tp, _Alloc >, [1978](#)
- std::experimental::fundamentals_v1::any, [1116](#)
- std::forward_list< _Tp, _Alloc >, [2082](#)
- std::list< _Tp, _Alloc >, [2294](#)
- std::map< _Key, _Tp, _Compare, _Alloc >, [2347](#)
- std::match_results< _Bi_iter, _Alloc >, [2371](#)
- std::multimap< _Key, _Tp, _Compare, _Alloc >, [2440](#)
- std::multiset< _Key, _Compare, _Alloc >, [2467](#)
- std::priority_queue< _Tp, _Sequence, _Compare >, [2643](#)
- std::queue< _Tp, _Sequence >, [2652](#)
- std::set< _Key, _Compare, _Alloc >, [2754](#)
- std::stack< _Tp, _Sequence >, [2803](#)
- std::tr2::dynamic_bitset< _WordT, _Alloc >, [2009](#)
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2993](#)
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3024](#)
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3052](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3078](#)
- std::vector< _Tp, _Alloc >, [3112](#)
- enable_if_t
- Metaprogramming, [398](#)
- enable_special_members.h, [3201](#)
- enable_view
- ranges_base.h, [3251](#)
- enc_filebuf.h, [3401](#)
- end
- __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [900](#), [901](#)
- __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, [2869](#)
- __gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp >, [1019](#)
- __gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >, [2932](#)
- __gnu_pbds::sample_trie_access_traits, [2729](#)
- __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >, [2937](#)
- File System, [168](#), [169](#)
- Numeric Arrays, [232](#)
- std, [650](#), [651](#)
- std::_Temporary_buffer< _ForwardIterator, _Tp >, [1083](#)
- std::basic_fstream< _CharT, _Traits >, [1237](#)
- std::basic_ifstream< _CharT, _Traits >, [1280](#)
- std::basic_ios< _CharT, _Traits >, [1303](#)
- std::basic_iostream< _CharT, _Traits >, [1349](#)
- std::basic_istream< _CharT, _Traits >, [1387](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1428](#)
- std::basic_ofstream< _CharT, _Traits >, [1462](#)
- std::basic_ostream< _CharT, _Traits >, [1494](#)
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1526](#)
- std::basic_string< _CharT, _Traits, _Alloc >, [1616](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1722](#)
- std::deque< _Tp, _Alloc >, [1978](#)
- std::forward_list< _Tp, _Alloc >, [2082](#)
- std::initializer_list< _E >, [2179](#)
- std::ios_base, [2199](#)
- std::list< _Tp, _Alloc >, [2294](#)
- std::map< _Key, _Tp, _Compare, _Alloc >, [2347](#)
- std::match_results< _Bi_iter, _Alloc >, [2371](#)
- std::multimap< _Key, _Tp, _Compare, _Alloc >, [2440](#)
- std::multiset< _Key, _Compare, _Alloc >, [2468](#)
- std::set< _Key, _Compare, _Alloc >, [2754](#)
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2993](#), [2994](#)
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3024](#), [3025](#)
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3052](#), [3053](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3078](#), [3080](#)
- std::vector< _Tp, _Alloc >, [3113](#)
- endian
- Bit manipulation, [184](#)
- endl
- std, [651](#)
- ends
- std, [651](#)
- entry_cmp.hpp, [3411](#)
- entry_list_fn_imps.hpp, [3423](#)
- entry_metadata_base.hpp, [3432](#)
- entry_pred.hpp, [3411](#)
- eof
- std::basic_fstream< _CharT, _Traits >, [1202](#)
- std::basic_ifstream< _CharT, _Traits >, [1252](#)
- std::basic_ios< _CharT, _Traits >, [1292](#)
- std::basic_iostream< _CharT, _Traits >, [1316](#)
- std::basic_istream< _CharT, _Traits >, [1361](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1399](#)
- std::basic_ofstream< _CharT, _Traits >, [1441](#)

- std::basic_ostream< _CharT, _Traits >, 1473
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1506
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1689
- eofbit
 - std::basic_fstream< _CharT, _Traits >, 1237
 - std::basic_ifstream< _CharT, _Traits >, 1280
 - std::basic_ios< _CharT, _Traits >, 1303
 - std::basic_iostream< _CharT, _Traits >, 1349
 - std::basic_istream< _CharT, _Traits >, 1387
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1428
 - std::basic_ofstream< _CharT, _Traits >, 1462
 - std::basic_ostream< _CharT, _Traits >, 1494
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1526
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1722
 - std::ios_base, 2199
- ep_ptr
 - __gnu_cxx::enc_filebuf< _CharT >, 2024
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2809
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2827
 - std::basic_filebuf< _CharT, _Traits >, 1175
 - std::basic_streambuf< _CharT, _Traits >, 1544
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1664
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3129
- epsilon
 - std::numeric_limits< _Tp >, 2529
- eq_by_less.hpp, 3426
- equal
 - Non-Mutating, 47–49
 - std::istreambuf_iterator< _CharT, _Traits >, 2255
- equal_range
 - Binary Search, 88
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2347, 2348
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2440–2442
 - std::multiset< _Key, _Compare, _Alloc >, 2468, 2469
 - std::set< _Key, _Compare, _Alloc >, 2754–2756
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2994, 2995
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3025, 3026
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3053, 3054
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3080, 3081
- equally_split.h, 3510
- equals
 - std::tr2::bool_set, 1776
- equivalent
 - std::V2::error_category, 2047
- erase
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 901, 902
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1572
 - std::basic_string< _CharT, _Traits, _Alloc >, 1616, 1617
 - std::deque< _Tp, _Alloc >, 1978
 - std::list< _Tp, _Alloc >, 2294
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2349, 2350
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2442, 2443
 - std::multiset< _Key, _Compare, _Alloc >, 2469, 2470
 - std::set< _Key, _Compare, _Alloc >, 2756, 2757
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2996, 2997
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3026, 3027
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3054, 3055
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3082, 3083
 - std::vector< _Tp, _Alloc >, 3113
- erase_after
 - std::forward_list< _Tp, _Alloc >, 2082, 2083
- erase_can_throw
 - __gnu_pbds::container_traits< Cntr >, 1878
- erase_fn_imps.hpp, 3411–3413
- erase_if.h, 3201
- erase_no_store_hash_fn_imps.hpp, 3424
- erase_store_hash_fn_imps.hpp, 3424
- error_backref
 - std::regex_constants, 776
- error_badbrace
 - std::regex_constants, 776
- error_badrepeat
 - std::regex_constants, 776
- error_brace
 - std::regex_constants, 776
- error_brack
 - std::regex_constants, 776
- error_collate
 - std::regex_constants, 777
- error_complexity
 - std::regex_constants, 777
- error_condition
 - std::error_condition, 2050
- error_constants.h, 3592
- error_ctype
 - std::regex_constants, 777
- error_escape

- std::regex_constants, 777
- error_paren
 - std::regex_constants, 777
- error_range
 - std::regex_constants, 777
- error_space
 - std::regex_constants, 777
- error_stack
 - std::regex_constants, 777
- error_type
 - std::regex_constants, 776
- event
 - std::basic_fstream< _CharT, _Traits >, 1199
 - std::basic_ifstream< _CharT, _Traits >, 1250
 - std::basic_ios< _CharT, _Traits >, 1290
 - std::basic_iostream< _CharT, _Traits >, 1314
 - std::basic_istream< _CharT, _Traits >, 1359
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1397
 - std::basic_ofstream< _CharT, _Traits >, 1438
 - std::basic_ostream< _CharT, _Traits >, 1471
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1504
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1686
 - std::ios_base, 2191
- event_callback
 - std::basic_fstream< _CharT, _Traits >, 1197
 - std::basic_ifstream< _CharT, _Traits >, 1248
 - std::basic_ios< _CharT, _Traits >, 1288
 - std::basic_iostream< _CharT, _Traits >, 1312
 - std::basic_istream< _CharT, _Traits >, 1357
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1395
 - std::basic_ofstream< _CharT, _Traits >, 1436
 - std::basic_ostream< _CharT, _Traits >, 1470
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1502
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1684
 - std::ios_base, 2190
- exception, 3152
- exception.h, 3202
- exception.hpp, 3443
- exception_defines.h, 3202
- exception_ptr.h, 3202
- Exceptions, 134, 150
 - __verbose_terminate_handler, 137
 - current_exception, 137
 - get_terminate, 137
 - get_unexpected, 137
 - make_exception_ptr, 137
 - rethrow_exception, 138
 - rethrow_if_nested, 138
 - set_terminate, 138
 - set_unexpected, 138
 - terminate, 138
 - terminate_handler, 136
 - throw_with_nested, 139
 - uncaught_exception, 139
 - uncaught_exceptions, 139
 - unexpected, 139
 - unexpected_handler, 136
- exceptions
 - std::basic_fstream< _CharT, _Traits >, 1202
 - std::basic_ifstream< _CharT, _Traits >, 1252, 1253
 - std::basic_ios< _CharT, _Traits >, 1292
 - std::basic_iostream< _CharT, _Traits >, 1316
 - std::basic_istream< _CharT, _Traits >, 1361
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1399
 - std::basic_ofstream< _CharT, _Traits >, 1441
 - std::basic_ostream< _CharT, _Traits >, 1473
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1506
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1689
- exchange
 - std, 651
- exclusive_scan
 - Generalized Numeric operations, 8, 9
- exp
 - Complex Numbers, 190
- expected, 3396
- experimental/memory_resource
 - get_default_resource, 3484
 - set_default_resource, 3484
- expint
 - Mathematical Special Functions, 216
 - TR1 Mathematical Special Functions, 265
- expintf
 - Mathematical Special Functions, 217
- expintl
 - Mathematical Special Functions, 217
- exponential_distribution
 - std::exponential_distribution< _RealType >, 2055
- extc++.h, 3593
- extended
 - std::regex_constants, 781
- Extensions, 140
- external_load_access
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, 1781
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, 2156
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, 2167

extptr_allocator.h, [3401](#)

extract

__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue,
_Compare, _Alloc >, [2676](#)

std::map< _Key, _Tp, _Compare, _Alloc >, [2350](#),
[2351](#)

std::multimap< _Key, _Tp, _Compare, _Alloc >, [2444](#)

std::multiset< _Key, _Compare, _Alloc >, [2471](#)

std::set< _Key, _Compare, _Alloc >, [2757](#)

std::unordered_map< _Key, _Tp, _Hash, _Pred, _Al-
loc >, [2997](#), [2998](#)

std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
_Alloc >, [3028](#)

std::unordered_multiset< _Value, _Hash, _Pred, _Al-
loc >, [3056](#)

std::unordered_set< _Value, _Hash, _Pred, _Alloc
>, [3083](#)

fabs

Complex Numbers, [190](#)

std, [651](#)

facet

std::locale::facet, [2062](#)

fail

std::basic_fstream< _CharT, _Traits >, [1203](#)

std::basic_ifstream< _CharT, _Traits >, [1253](#)

std::basic_ios< _CharT, _Traits >, [1293](#)

std::basic_iostream< _CharT, _Traits >, [1317](#)

std::basic_istream< _CharT, _Traits >, [1362](#)

std::basic_istreamstream< _CharT, _Traits, _Alloc >,
[1401](#)

std::basic_ofstream< _CharT, _Traits >, [1442](#)

std::basic_ostream< _CharT, _Traits >, [1475](#)

std::basic_ostreamstream< _CharT, _Traits, _Alloc >,
[1507](#)

std::basic_stringstream< _CharT, _Traits, _Alloc >,
[1690](#)

failbit

std::basic_fstream< _CharT, _Traits >, [1237](#)

std::basic_ifstream< _CharT, _Traits >, [1280](#)

std::basic_ios< _CharT, _Traits >, [1304](#)

std::basic_iostream< _CharT, _Traits >, [1349](#)

std::basic_istream< _CharT, _Traits >, [1387](#)

std::basic_istreamstream< _CharT, _Traits, _Alloc >,
[1428](#)

std::basic_ofstream< _CharT, _Traits >, [1462](#)

std::basic_ostream< _CharT, _Traits >, [1494](#)

std::basic_ostreamstream< _CharT, _Traits, _Alloc >,
[1526](#)

std::basic_stringstream< _CharT, _Traits, _Alloc >,
[1722](#)

std::ios_base, [2199](#)

failed

std::ostreambuf_iterator< _CharT, _Traits >, [2574](#)

false_type

Metaprogramming, [398](#)

false_name

std::numpunct< _CharT >, [2556](#)

std::numpunct_byname< _CharT >, [2560](#)

fd

__gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2809](#)

features.h, [3511](#)

_GLIBCXX_BAL_QUICKSORT, [3511](#)

_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS,
[3511](#)

_GLIBCXX_FIND_EQUAL_SPLIT, [3511](#)

_GLIBCXX_FIND_GROWING_BLOCKS, [3512](#)

_GLIBCXX_MERGESORT, [3512](#)

_GLIBCXX_QUICKSORT, [3512](#)

_GLIBCXX_TREE_DYNAMIC_BALANCING, [3512](#)

_GLIBCXX_TREE_FULL_COPY, [3512](#)

_GLIBCXX_TREE_INITIAL_SPLITTING, [3512](#)

fcntl.h, [3462](#)

file

__gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2809](#)

__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >,
[2828](#)

File System, [163](#)

begin, [168](#)

copy_options, [168](#)

directory_options, [168](#)

end, [168](#), [169](#)

file_time_type, [168](#)

file_type, [168](#)

perm_options, [168](#)

perms, [168](#)

u8path, [169](#)

file_time_type

File System, [168](#)

Filesystem TS, [308](#)

file_type

File System, [168](#)

Filesystem TS, [308](#)

filebuf

I/O, [171](#)

filesystem, [3463](#)

Filesystem TS, [303](#)

copy_options, [308](#)

directory_options, [308](#)

file_time_type, [308](#)

file_type, [308](#)

perms, [308](#)

fill

Mutating, [25](#)

std::basic_fstream< _CharT, _Traits >, [1203](#)

std::basic_ifstream< _CharT, _Traits >, [1253](#), [1254](#)

std::basic_ios< _CharT, _Traits >, [1293](#)

- std::basic_iostream< _CharT, _Traits >, [1317](#)
- std::basic_istream< _CharT, _Traits >, [1362](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1401](#)
- std::basic_ofstream< _CharT, _Traits >, [1442](#)
- std::basic_ostream< _CharT, _Traits >, [1475](#)
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1507](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1690](#)
- fill_minimal_n
 - __gnu_parallel::Settings, [1075](#)
- fill_n
 - Mutating, [25](#)
- find
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [902](#), [903](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1572](#), [1573](#)
 - Non-Mutating, [49](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1618](#), [1619](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2351](#), [2352](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2444](#), [2445](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2471](#), [2472](#)
 - std::set< _Key, _Compare, _Alloc >, [2757](#)–[2759](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2998](#), [2999](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3028](#), [3029](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3056](#), [3057](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3083](#)–[3085](#)
- find.h, [3513](#)
- find_by_order
 - __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >, [2917](#)
 - __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, ATraits, _Alloc >, [2930](#)
- find_end
 - Non-Mutating, [50](#), [51](#)
- find_first
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2009](#)
- find_first_not_of
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [904](#), [905](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1573](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1620](#), [1621](#)
- find_first_of
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [905](#)–[907](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1574](#)
 - Non-Mutating, [51](#), [52](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1623](#), [1624](#)
- find_fn_imps.hpp, [3413](#), [3414](#)
- find_if
 - Non-Mutating, [52](#)
- find_if_not
 - Non-Mutating, [53](#)
- find_increasing_factor
 - __gnu_parallel::Settings, [1075](#)
- find_initial_block_size
 - __gnu_parallel::Settings, [1075](#)
- find_last_not_of
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [907](#)–[909](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1575](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1626](#), [1627](#)
- find_last_of
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [909](#), [910](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1575](#), [1576](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1629](#), [1630](#)
- find_maximum_block_size
 - __gnu_parallel::Settings, [1075](#)
- find_next
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2010](#)
- find_no_store_hash_fn_imps.hpp, [3427](#)
- find_scale_factor
 - __gnu_parallel::Settings, [1075](#)
- find_selectors.h, [3513](#)
- find_sequential_search_size
 - __gnu_parallel::Settings, [1075](#)
- find_store_hash_fn_imps.hpp, [3424](#)
- first
 - __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [969](#)
 - std::pair< _T1, _T2 >, [2593](#)
 - std::sub_match< _Biter >, [2857](#)
- first_argument_type
 - __gnu_cxx::project1st< _Arg1, _Arg2 >, [2645](#)
 - __gnu_cxx::project2nd< _Arg1, _Arg2 >, [2646](#)
 - __gnu_parallel::EqualFromLess< _T1, _T2, _Compare >, [950](#)
 - __gnu_parallel::EqualTo< _T1, _T2 >, [951](#)
 - __gnu_parallel::Less< _T1, _T2 >, [973](#)

- `__gnu_parallel::Lexicographic< _T1, _T2, _Compare >`, [974](#)
- `__gnu_parallel::LexicographicReverse< _T1, _T2, _Compare >`, [975](#)
- `__gnu_parallel::Multiplies< _Tp1, _Tp2, _Result >`, [999](#)
- `__gnu_parallel::Plus< _Tp1, _Tp2, _Result >`, [1014](#)
- `std::binary_function< _Arg1, _Arg2, _Result >`, [1741](#)
- `std::binary_negate< _Predicate >`, [1751](#)
- `std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >`, [1861](#)
- `std::const_mem_fun1_t< _Ret, _Tp, _Arg >`, [1862](#)
- `std::divides< _Tp >`, [2000](#)
- `std::equal_to< _Tp >`, [2045](#)
- `std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >`, [2585](#)
- `std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >`, [2588](#)
- `std::greater< _Tp >`, [2126](#)
- `std::greater_equal< _Tp >`, [2128](#)
- `std::less< _Tp >`, [2272](#)
- `std::less_equal< _Tp >`, [2274](#)
- `std::logical_and< _Tp >`, [2319](#)
- `std::logical_or< _Tp >`, [2322](#)
- `std::mem_fun1_ref_t< _Ret, _Tp, _Arg >`, [2377](#)
- `std::mem_fun1_t< _Ret, _Tp, _Arg >`, [2379](#)
- `std::minus< _Tp >`, [2392](#)
- `std::modulus< _Tp >`, [2394](#)
- `std::multiplies< _Tp >`, [2455](#)
- `std::not_equal_to< _Tp >`, [2496](#)
- `std::owner_less< shared_ptr< _Tp > >`, [2586](#)
- `std::owner_less< void >`, [2587](#)
- `std::owner_less< weak_ptr< _Tp > >`, [2588](#)
- `std::plus< _Tp >`, [2628](#)
- `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >`, [2630](#)
- first_type
 - `__gnu_parallel::IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >`, [967](#)
 - `std::pair< _T1, _T2 >`, [2592](#)
 - `std::sub_match< _Biter >`, [2853](#)
- fixed
 - `std`, [651](#)
 - `std::basic_fstream< _CharT, _Traits >`, [1237](#)
 - `std::basic_ifstream< _CharT, _Traits >`, [1281](#)
 - `std::basic_ios< _CharT, _Traits >`, [1304](#)
 - `std::basic_iostream< _CharT, _Traits >`, [1350](#)
 - `std::basic_istream< _CharT, _Traits >`, [1387](#)
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, [1429](#)
 - `std::basic_ofstream< _CharT, _Traits >`, [1462](#)
 - `std::basic_ostream< _CharT, _Traits >`, [1494](#)
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, [1527](#)
- `std::basic_stringstream< _CharT, _Traits, _Alloc >`, [1723](#)
- `std::ios_base`, [2199](#)
- flags
 - `std::basic_fstream< _CharT, _Traits >`, [1203](#), [1204](#)
 - `std::basic_ifstream< _CharT, _Traits >`, [1254](#)
 - `std::basic_ios< _CharT, _Traits >`, [1294](#)
 - `std::basic_iostream< _CharT, _Traits >`, [1317](#), [1318](#)
 - `std::basic_istream< _CharT, _Traits >`, [1362](#), [1363](#)
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, [1402](#)
 - `std::basic_ofstream< _CharT, _Traits >`, [1443](#)
 - `std::basic_ostream< _CharT, _Traits >`, [1476](#)
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, [1508](#)
 - `std::basic_regex< _Ch_type, _Rx_traits >`, [1536](#)
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, [1690](#), [1691](#)
 - `std::ios_base`, [2192](#)
- flip
 - `std::bitset< _Nb >`, [1770](#)
 - `std::tr2::dynamic_bitset< _WordT, _Alloc >`, [2010](#)
- float_denorm_style
 - `std`, [634](#)
- float_round_style
 - `std`, [635](#)
- floatfield
 - `std::basic_fstream< _CharT, _Traits >`, [1237](#)
 - `std::basic_ifstream< _CharT, _Traits >`, [1281](#)
 - `std::basic_ios< _CharT, _Traits >`, [1304](#)
 - `std::basic_iostream< _CharT, _Traits >`, [1350](#)
 - `std::basic_istream< _CharT, _Traits >`, [1387](#)
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, [1429](#)
 - `std::basic_ofstream< _CharT, _Traits >`, [1463](#)
 - `std::basic_ostream< _CharT, _Traits >`, [1495](#)
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, [1527](#)
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, [1723](#)
 - `std::ios_base`, [2199](#)
- floor
 - `Time`, [416](#)
- flush
 - `std`, [652](#)
 - `std::basic_fstream< _CharT, _Traits >`, [1204](#)
 - `std::basic_iostream< _CharT, _Traits >`, [1318](#)
 - `std::basic_ofstream< _CharT, _Traits >`, [1443](#)
 - `std::basic_ostream< _CharT, _Traits >`, [1476](#)
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, [1508](#)
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, [1691](#)
- fmtflags

- std::basic_fstream< _CharT, _Traits >, 1197
- std::basic_ifstream< _CharT, _Traits >, 1248
- std::basic_ios< _CharT, _Traits >, 1288
- std::basic_iostream< _CharT, _Traits >, 1312
- std::basic_istream< _CharT, _Traits >, 1358
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1395
- std::basic_ofstream< _CharT, _Traits >, 1437
- std::basic_ostream< _CharT, _Traits >, 1470
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1502
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1685
- std::ios_base, 2190
- for_each
 - Non-Mutating, 53
- for_each.h, 3514
- for_each_minimal_n
 - __gnu_parallel::Settings, 1076
- for_each_n
 - Non-Mutating, 54
- for_each_selectors.h, 3514
- format
 - std::filesystem::path, 2619
 - std::match_results< _Bi_iter, _Alloc >, 2371, 2372
- format_default
 - std::regex_constants, 781
- format_first_only
 - std::regex_constants, 781
- format_no_copy
 - std::regex_constants, 782
- format_sed
 - std::regex_constants, 782
- formatter.h, 3369
- forward
 - Utilities, 344
- forward_as_tuple
 - Utilities, 344
- forward_list, 3463, 3464
 - std::forward_list< _Tp, _Alloc >, 2076–2078
- forward_list.h, 3203
- forward_list.tcc, 3203
- fpos
 - std::fpos< _StateT >, 2092
- frac_digits
 - std::moneypunct< _CharT, _Intl >, 2412
 - std::moneypunct_byname< _CharT, _Intl >, 2419
- from_bytes
 - std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >, 3147
- from_chars
 - std, 652
- front
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 911
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1576
 - std::basic_string< _CharT, _Traits, _Alloc >, 1632
 - std::deque< _Tp, _Alloc >, 1980
 - std::forward_list< _Tp, _Alloc >, 2083
 - std::list< _Tp, _Alloc >, 2295
 - std::queue< _Tp, _Sequence >, 2652
 - std::vector< _Tp, _Alloc >, 3114
- front_insert_iterator
 - std::front_insert_iterator< _Container >, 2096
- front_inserter
 - Iterators, 177
- fs_dir.h, 3204
- fs_fwd.h, 3205, 3207
- fs_ops.h, 3208, 3211
- fs_path.h, 3213
- fstream, 3465
 - I/O, 171
- fstream.tcc, 3214
- functexcept.h, 3214
- function
 - std::function< _Res(_ArgTypes...) >, 2098, 2099
- Function Objects, 351
 - mem_fn, 352
 - not_fn, 352
- functional, 3466–3468
- functional_hash.h, 3215
- functions.h, 3369
- future, 3470
 - std::future< _Res >, 2104
 - std::future< _Res & >, 2106
 - std::future< void >, 2107
- future_category
 - Futures, 121
- future_errc
 - Futures, 120
- future_status
 - Futures, 120
- Futures, 119
 - async, 121
 - future_category, 121
 - future_errc, 120
 - future_status, 120
 - launch, 121
 - make_error_code, 121
 - make_error_condition, 121
 - swap, 122
- gamma_distribution
 - std::gamma_distribution< _RealType >, 2110
- gbump
 - __gnu_cxx::enc_filebuf< _CharT >, 2025

- `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2809
- `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2828
- `std::basic_filebuf< _CharT, _Traits >`, 1175
- `std::basic_streambuf< _CharT, _Traits >`, 1544
- `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1664
- `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3129
- gcd
 - `std`, 652
 - `std::experimental`, 764
- gcount
 - `std::basic_fstream< _CharT, _Traits >`, 1204
 - `std::basic_ifstream< _CharT, _Traits >`, 1255
 - `std::basic_iostream< _CharT, _Traits >`, 1318
 - `std::basic_istream< _CharT, _Traits >`, 1363
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1402
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1691
- Generalized Numeric operations, 4
 - `accumulate`, 6
 - `adjacent_difference`, 7
 - `exclusive_scan`, 8, 9
 - `inclusive_scan`, 9, 10
 - `inner_product`, 11, 12
 - `iota`, 12
 - `partial_sum`, 13
 - `reduce`, 15, 16
 - `transform_exclusive_scan`, 16
 - `transform_inclusive_scan`, 17, 18
 - `transform_reduce`, 18, 19
- generate
 - `Mutating`, 26
- generate_canonical
 - Random Number Generation, 242
- generate_minimal_n
 - `__gnu_parallel::Settings`, 1076
- generate_n
 - `Mutating`, 26
- generic_category
 - Diagnostics, 131
- get
 - `__gnu_parallel::Settings`, 1074
 - `std`, 652–654
 - `std::auto_ptr< _Tp >`, 1154
 - `std::basic_fstream< _CharT, _Traits >`, 1204–1206
 - `std::basic_ifstream< _CharT, _Traits >`, 1255–1257
 - `std::basic_iostream< _CharT, _Traits >`, 1318–1320
 - `std::basic_istream< _CharT, _Traits >`, 1363–1365
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1402–1404
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1691–1693
 - `std::future< _Res >`, 2104
 - `std::future< _Res & >`, 2106
 - `std::future< void >`, 2107
 - `std::money_get< _CharT, _InIter >`, 2399, 2400
 - `std::num_get< _CharT, _InIter >`, 2508–2514
 - `std::shared_future< _Res >`, 2767
 - `std::shared_future< _Res & >`, 2769
 - `std::shared_ptr< _Tp >`, 2783
 - `std::time_get< _CharT, _InIter >`, 2888, 2889
 - `std::time_get_byname< _CharT, _InIter >`, 2898, 2899
 - `std::unique_ptr< _Tp, _Dp >`, 2967
 - `std::unique_ptr< _Tp[], _Dp >`, 2972
 - Utilities, 344, 345
- get_actual_size
 - `__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >`, 2167
- get_allocator
 - `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >`, 911
 - `__gnu_debug::basic_string< _CharT, _Traits, _Allocator >`, 1577
 - `std::basic_string< _CharT, _Traits, _Alloc >`, 1632
 - `std::deque< _Tp, _Alloc >`, 1980
 - `std::forward_list< _Tp, _Alloc >`, 2083
 - `std::list< _Tp, _Alloc >`, 2295
 - `std::map< _Key, _Tp, _Compare, _Alloc >`, 2352
 - `std::match_results< _Bi_iter, _Alloc >`, 2372
 - `std::multimap< _Key, _Tp, _Compare, _Alloc >`, 2445
 - `std::multiset< _Key, _Compare, _Alloc >`, 2472
 - `std::set< _Key, _Compare, _Alloc >`, 2759
 - `std::tr2::dynamic_bitset< _WordT, _Alloc >`, 2011
 - `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >`, 2999
 - `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3030
 - `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`, 3058
 - `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`, 3085
 - `std::vector< _Tp, _Alloc >`, 3114
- get_child
 - `__gnu_pbds::detail::pat_trie_base::Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >`, 1002
 - `__gnu_pbds::detail::pat_trie_base::Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >`, 1008
- get_comb_hash_fn
 - `__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >`, 1792
- get_comb_probe_fn

- __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2124](#)
- get_date
 - std::time_get< _CharT, _InIter >, [2889](#)
 - std::time_get_byname< _CharT, _InIter >, [2900](#)
- get_default_resource
 - experimental/memory_resource, [3484](#)
 - memory_resource, [3485](#)
- get_deleter
 - Pointer Abstractions, [376](#)
 - std::experimental, [764](#)
 - std::unique_ptr< _Tp, _Dp >, [2968](#)
 - std::unique_ptr< _Tp[], _Dp >, [2973](#)
- get_eq_fn
 - __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >, [1793](#)
 - __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2124](#)
- get_hash_fn
 - __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >, [1793](#)
 - __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2124](#)
- get_id
 - std::this_thread, [786](#)
- get_l_child
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1732](#)
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1736](#)
 - __gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc >, [2580](#)
 - __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc >, [2582](#)
- get_load
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1782](#)
- get_loads
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2157](#)
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2167](#)
- get_metadata
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1732](#)
- __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1736](#)
 - __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, [1002](#)
 - __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, [1008](#)
- get_money
 - std, [654](#)
- get_monthname
 - std::time_get< _CharT, _InIter >, [2890](#)
 - std::time_get_byname< _CharT, _InIter >, [2900](#)
- get_nearest_larger_size
 - __gnu_pbds::sample_size_policy, [2728](#)
- get_nearest_smaller_size
 - __gnu_pbds::sample_size_policy, [2728](#)
- get_new_handler
 - std, [654](#)
- get_new_size
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2167](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
- get_pointer_safety
 - Pointer Safety and Garbage Collection, [388](#)
- get_probe_fn
 - __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2124](#), [2125](#)
- get_r_child
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1732](#)
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1737](#)
 - __gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc >, [2580](#)
 - __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc >, [2582](#)
- get_resize_policy
 - __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >, [1793](#)
 - __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >, [2125](#)
- get_size_policy
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2168](#)
- get_temporary_buffer
 - std, [654](#)

- get_terminate
 - Exceptions, [137](#)
- get_time
 - std, [655](#)
 - std::time_get< _CharT, _InIter >, [2891](#)
 - std::time_get_byname< _CharT, _InIter >, [2901](#)
- get_trigger_policy
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2168](#)
- get_unexpected
 - Exceptions, [137](#)
- get_weekday
 - std::time_get< _CharT, _InIter >, [2891](#)
 - std::time_get_byname< _CharT, _InIter >, [2901](#)
- get_year
 - std::time_get< _CharT, _InIter >, [2892](#)
 - std::time_get_byname< _CharT, _InIter >, [2902](#)
- getline
 - std, [655–657](#)
 - std::basic_fstream< _CharT, _Traits >, [1207](#), [1208](#)
 - std::basic_ifstream< _CharT, _Traits >, [1257](#), [1258](#)
 - std::basic_iostream< _CharT, _Traits >, [1321](#), [1322](#)
 - std::basic_istream< _CharT, _Traits >, [1366](#), [1367](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1405](#), [1406](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1694](#), [1695](#)
- getloc
 - __gnu_cxx::enc_filebuf< _CharT >, [2025](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2810](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2828](#)
 - std::basic_filebuf< _CharT, _Traits >, [1175](#)
 - std::basic_fstream< _CharT, _Traits >, [1208](#)
 - std::basic_ifstream< _CharT, _Traits >, [1258](#)
 - std::basic_ios< _CharT, _Traits >, [1294](#)
 - std::basic_iostream< _CharT, _Traits >, [1322](#)
 - std::basic_istream< _CharT, _Traits >, [1367](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1406](#)
 - std::basic_ofstream< _CharT, _Traits >, [1444](#)
 - std::basic_ostream< _CharT, _Traits >, [1476](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1508](#)
 - std::basic_regex< _Ch_type, _Rx_traits >, [1536](#)
 - std::basic_streambuf< _CharT, _Traits >, [1544](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1665](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1695](#)
 - std::ios_base, [2192](#)
 - std::regex_traits< _Ch_type >, [2699](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3130](#)
- global
 - std::locale, [2313](#)
- good
 - std::basic_fstream< _CharT, _Traits >, [1208](#)
 - std::basic_ifstream< _CharT, _Traits >, [1259](#)
 - std::basic_ios< _CharT, _Traits >, [1294](#)
 - std::basic_iostream< _CharT, _Traits >, [1322](#)
 - std::basic_istream< _CharT, _Traits >, [1367](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1406](#)
 - std::basic_ofstream< _CharT, _Traits >, [1444](#)
 - std::basic_ostream< _CharT, _Traits >, [1477](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1509](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1695](#)
- goodbit
 - std::basic_fstream< _CharT, _Traits >, [1238](#)
 - std::basic_ifstream< _CharT, _Traits >, [1281](#)
 - std::basic_ios< _CharT, _Traits >, [1304](#)
 - std::basic_iostream< _CharT, _Traits >, [1350](#)
 - std::basic_istream< _CharT, _Traits >, [1387](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1429](#)
 - std::basic_ofstream< _CharT, _Traits >, [1463](#)
 - std::basic_ostream< _CharT, _Traits >, [1495](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1527](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1723](#)
 - std::ios_base, [2200](#)
- gp_hash_table
 - __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >, [2116–2119](#)
- gp_ht_map.hpp, [3427](#)
- gptr
 - __gnu_cxx::enc_filebuf< _CharT >, [2025](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2810](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2828](#)
 - std::basic_filebuf< _CharT, _Traits >, [1176](#)
 - std::basic_streambuf< _CharT, _Traits >, [1544](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1665](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3130](#)
- grep
 - std::regex_constants, [782](#)
- grouping
 - std::moneypunct< _CharT, _Intl >, [2412](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2420](#)
 - std::numpunct< _CharT >, [2556](#)
 - std::numpunct_byname< _CharT >, [2560](#)
- gslice
 - Numeric Arrays, [227](#), [228](#)

- gslice.h, [3215](#)
- gslice_array
 - Numeric Arrays, [228](#)
- gslice_array.h, [3216](#)
- has_denorm
 - std::__numeric_limits_base, [863](#)
 - std::numeric_limits< _Tp >, [2531](#)
- has_denorm_loss
 - std::__numeric_limits_base, [863](#)
 - std::numeric_limits< _Tp >, [2531](#)
- has_facet
 - Locales, [181](#)
 - std::locale, [2314](#)
 - std::locale::id, [2169](#)
- has_infinity
 - std::__numeric_limits_base, [864](#)
 - std::numeric_limits< _Tp >, [2531](#)
- has_quiet_NaN
 - std::__numeric_limits_base, [864](#)
 - std::numeric_limits< _Tp >, [2531](#)
- has_signaling_NaN
 - std::__numeric_limits_base, [864](#)
 - std::numeric_limits< _Tp >, [2531](#)
- has_single_bit
 - Bit manipulation, [185](#)
- has_value
 - std::any, [1114](#)
- hash
 - std::collate< _CharT >, [1843](#)
 - std::collate_byname< _CharT >, [1847](#)
- Hash-Based, [148](#)
- hash_bytes.h, [3216](#)
- hash_eq_fn.hpp, [3427](#)
- hash_exponential_size_policy
 - __gnu_pbds::hash_exponential_size_policy< Size_Type >, [2155](#)
- hash_exponential_size_policy_imp.hpp, [3437](#)
- hash_fun.h, [3165](#)
- hash_function
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2999](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3030](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3058](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3085](#)
- hash_load_check_resize_trigger
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2156](#)
- hash_load_check_resize_trigger_imp.hpp, [3437](#)
- hash_load_check_resize_trigger_size_base.hpp, [3437](#)
- hash_map, [3165](#)
- hash_policy.hpp, [3444](#)
- hash_prime_size_policy
 - __gnu_pbds::hash_prime_size_policy, [2163](#)
- hash_prime_size_policy_imp.hpp, [3437](#)
- hash_set, [3166](#)
- hash_standard_resize_policy
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2167](#)
- hash_standard_resize_policy_imp.hpp, [3437](#)
- hasher
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- Hashes, [359](#)
- hashtable.h, [3216](#), [3217](#)
- hashtable_policy.h, [3217](#)
- Heap, [91](#)
 - is_heap, [92](#)
 - is_heap_until, [93](#)
 - make_heap, [94](#)
 - pop_heap, [95](#)
 - push_heap, [95](#), [96](#)
 - sort_heap, [96](#), [97](#)
- Heap-Based, [149](#)
- helper_functions.h, [3371](#)
- hermite
 - Mathematical Special Functions, [217](#)
 - TR1 Mathematical Special Functions, [265](#)
- hermitef
 - Mathematical Special Functions, [217](#)
- hermitel
 - Mathematical Special Functions, [218](#)
- hex
 - std, [657](#)
 - std::basic_fstream< _CharT, _Traits >, [1238](#)
 - std::basic_ifstream< _CharT, _Traits >, [1281](#)
 - std::basic_ios< _CharT, _Traits >, [1304](#)
 - std::basic_iostream< _CharT, _Traits >, [1350](#)
 - std::basic_istream< _CharT, _Traits >, [1388](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1429](#)
 - std::basic_ofstream< _CharT, _Traits >, [1463](#)
 - std::basic_ostream< _CharT, _Traits >, [1495](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1527](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1723](#)
 - std::ios_base, [2200](#)

- hexfloat
 - std, [657](#)
- high_resolution_clock
 - Time, [413](#)
- hours
 - Time, [413](#)
- hyperg
 - Mathematical Special Functions, [218](#)
 - TR1 Mathematical Special Functions, [265](#)
- hypergf
 - Mathematical Special Functions, [218](#)
- hypergl
 - Mathematical Special Functions, [219](#)
- I/O, [169](#)
 - filebuf, [171](#)
 - fstream, [171](#)
 - ifstream, [171](#)
 - ios, [171](#)
 - iostream, [171](#)
 - istream, [171](#)
 - istringstream, [171](#)
 - ofstream, [171](#)
 - ostream, [171](#)
 - ostreamstream, [171](#)
 - streambuf, [171](#)
 - stringbuf, [171](#)
 - stringstream, [172](#)
 - wfilebuf, [172](#)
 - wfstream, [172](#)
 - wifstream, [172](#)
 - wios, [172](#)
 - wiostream, [172](#)
 - wistream, [172](#)
 - wistringstream, [172](#)
 - wofstream, [172](#)
 - wostream, [172](#)
 - wostreamstream, [172](#)
 - wstreambuf, [173](#)
 - wstringbuf, [173](#)
 - wstringstream, [173](#)
- icase
 - std::regex_constants, [782](#)
- id
 - std::collate< _CharT >, [1844](#)
 - std::collate_byname< _CharT >, [1848](#)
 - std::ctype< _CharT >, [1898](#)
 - std::ctype< char >, [1909](#)
 - std::ctype< wchar_t >, [1923](#)
 - std::ctype_byname< _CharT >, [1937](#)
 - std::ctype_byname< char >, [1948](#)
 - std::locale::id, [2169](#)
 - std::messages< _CharT >, [2389](#)
 - std::messages_byname< _CharT >, [2391](#)
 - std::money_get< _CharT, _InIter >, [2401](#)
 - std::money_put< _CharT, _OutIter >, [2405](#)
 - std::moneypunct< _CharT, _Intl >, [2414](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2422](#)
 - std::num_get< _CharT, _InIter >, [2515](#)
 - std::num_put< _CharT, _OutIter >, [2528](#)
 - std::numpunct< _CharT >, [2557](#)
 - std::numpunct_byname< _CharT >, [2561](#)
 - std::time_get< _CharT, _InIter >, [2892](#)
 - std::time_get_byname< _CharT, _InIter >, [2903](#)
 - std::time_put< _CharT, _OutIter >, [2907](#)
 - std::time_put_byname< _CharT, _OutIter >, [2911](#)
- identity_element
 - SGL, [161](#)
- ifstream
 - I/O, [171](#)
- ignore
 - std::basic_fstream< _CharT, _Traits >, [1208](#), [1209](#)
 - std::basic_ifstream< _CharT, _Traits >, [1259](#)
 - std::basic_iostream< _CharT, _Traits >, [1322](#), [1323](#)
 - std::basic_istream< _CharT, _Traits >, [1367](#), [1368](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1406](#), [1407](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1695](#), [1696](#)
- imbue
 - __gnu_cxx::enc_filebuf< _CharT >, [2025](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2810](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2829](#)
 - std::basic_filebuf< _CharT, _Traits >, [1176](#)
 - std::basic_fstream< _CharT, _Traits >, [1209](#)
 - std::basic_ifstream< _CharT, _Traits >, [1260](#)
 - std::basic_ios< _CharT, _Traits >, [1295](#)
 - std::basic_iostream< _CharT, _Traits >, [1323](#)
 - std::basic_istream< _CharT, _Traits >, [1368](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1407](#)
 - std::basic_ofstream< _CharT, _Traits >, [1444](#)
 - std::basic_ostream< _CharT, _Traits >, [1477](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1509](#)
 - std::basic_regex< _Ch_type, _Rx_traits >, [1536](#)
 - std::basic_streambuf< _CharT, _Traits >, [1544](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1665](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1696](#)
 - std::ios_base, [2193](#)
 - std::regex_traits< _Ch_type >, [2699](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3130](#)
- in
 - std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >, [819](#)
 - std::basic_fstream< _CharT, _Traits >, [1238](#)

- std::basic_ifstream< _CharT, _Traits >, [1281](#)
- std::basic_ios< _CharT, _Traits >, [1304](#)
- std::basic_iostream< _CharT, _Traits >, [1350](#)
- std::basic_istream< _CharT, _Traits >, [1388](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1429](#)
- std::basic_ofstream< _CharT, _Traits >, [1463](#)
- std::basic_ostream< _CharT, _Traits >, [1495](#)
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1527](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1723](#)
- std::codecvt< _InternT, _ExternT, _StateT >, [1805](#)
- std::codecvt< _InternT, _ExternT, encoding_state >, [1810](#)
- std::codecvt< char, char, mbstate_t >, [1815](#)
- std::codecvt< char16_t, char, mbstate_t >, [1820](#)
- std::codecvt< char32_t, char, mbstate_t >, [1824](#)
- std::codecvt< wchar_t, char, mbstate_t >, [1829](#)
- std::codecvt_byname< _InternT, _ExternT, _StateT >, [1835](#)
- std::ios_base, [2200](#)
- in_avail
 - __gnu_cxx::enc_filebuf< _CharT >, [2026](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2811](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2829](#)
 - std::basic_filebuf< _CharT, _Traits >, [1176](#)
 - std::basic_streambuf< _CharT, _Traits >, [1545](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1665](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3130](#)
- in_place
 - Optional values, [316](#)
- includes
 - Set Operations, [98, 99](#)
- inclusive_scan
 - Generalized Numeric operations, [9, 10](#)
- increment
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2281](#)
- independent_bits_engine
 - std::independent_bits_engine< __RandomNumberEngine, __w, _UIntType >, [2173](#)
- index_sequence
 - std, [633](#)
- index_sequence_for
 - std, [633](#)
- indirect_array
 - Numeric Arrays, [228](#)
- indirect_array.h, [3217](#)
- infinity
 - std::numeric_limits< _Tp >, [2530](#)
- info_fn_imps.hpp, [3414, 3415](#)
- init
 - std::basic_fstream< _CharT, _Traits >, [1210](#)
 - std::basic_ifstream< _CharT, _Traits >, [1260](#)
 - std::basic_ios< _CharT, _Traits >, [1295](#)
 - std::basic_iostream< _CharT, _Traits >, [1324](#)
 - std::basic_istream< _CharT, _Traits >, [1369](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1408](#)
 - std::basic_ofstream< _CharT, _Traits >, [1444](#)
 - std::basic_ostream< _CharT, _Traits >, [1477](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1509](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1697](#)
- initializer_list, [3152](#)
- inner_product
 - Generalized Numeric operations, [11, 12](#)
- inplace_merge
 - Sorting, [65, 66](#)
- insert
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [911–916](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1577–1580](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1632–1637](#)
 - std::deque< _Tp, _Alloc >, [1980–1982](#)
 - std::list< _Tp, _Alloc >, [2295–2297](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2352–2356](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2446–2449](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2473, 2474](#)
 - std::set< _Key, _Compare, _Alloc >, [2759–2761](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2999–3003](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3030–3033](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3058–3060](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3085–3088](#)
 - std::vector< _Tp, _Alloc >, [3114–3116](#)
- insert_after
 - std::forward_list< _Tp, _Alloc >, [2083–2085](#)
- insert_fn_imps.hpp, [3415, 3416](#)
- insert_iterator
 - std::insert_iterator< _Container >, [2183](#)
- insert_join_fn_imps.hpp, [3434](#)
- insert_no_store_hash_fn_imps.hpp, [3424](#)
- insert_or_assign
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2356, 2357](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3003, 3004](#)

- insert_store_hash_fn_imps.hpp, [3424](#)
- inserter
 - Iterators, [177](#)
- int_type
 - std::basic_ios< _CharT, _Traits >, [1289](#)
 - std::basic_streambuf< _CharT, _Traits >, [1542](#)
 - std::istreambuf_iterator< _CharT, _Traits >, [2254](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3127](#)
- internal
 - std, [657](#)
 - std::basic_fstream< _CharT, _Traits >, [1238](#)
 - std::basic_ifstream< _CharT, _Traits >, [1281](#)
 - std::basic_ios< _CharT, _Traits >, [1305](#)
 - std::basic_iostream< _CharT, _Traits >, [1351](#)
 - std::basic_istream< _CharT, _Traits >, [1388](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1429](#)
 - std::basic_ofstream< _CharT, _Traits >, [1463](#)
 - std::basic_ostream< _CharT, _Traits >, [1495](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1527](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1724](#)
 - std::ios_base, [2200](#)
- intervals
 - std::piecewise_constant_distribution< _RealType >, [2621](#)
 - std::piecewise_linear_distribution< _RealType >, [2625](#)
- intl
 - std::moneypunct< _CharT, _Intl >, [2414](#)
- inv_pi_v
 - std::numbers, [772](#)
- inv_sqrt3_v
 - std::numbers, [772](#)
- inv_sqrtpi_v
 - std::numbers, [772](#)
- Invalidation Guarantees, [153](#)
- invoke
 - std, [657](#)
- invoke.h, [3217](#)
- invoke_result_t
 - Metaprogramming, [398](#)
- io_errc
 - std, [635](#)
- iomanip, [3471](#)
- ios, [3472](#)
 - I/O, [171](#)
- ios_base.h, [3218](#)
- iosfwd, [3473](#)
- iostate
 - std::basic_fstream< _CharT, _Traits >, [1198](#)
 - std::basic_ifstream< _CharT, _Traits >, [1249](#)
 - std::basic_ios< _CharT, _Traits >, [1289](#)
 - std::basic_iostream< _CharT, _Traits >, [1313](#)
 - std::basic_istream< _CharT, _Traits >, [1358](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1396](#)
 - std::basic_ofstream< _CharT, _Traits >, [1437](#)
 - std::basic_ostream< _CharT, _Traits >, [1471](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1503](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1685](#)
 - std::ios_base, [2190](#)
- iostream, [3473](#)
 - I/O, [171](#)
- iota
 - Generalized Numeric operations, [12](#)
- is
 - std::__ctype_abstract_base< _CharT >, [832](#)
 - std::ctype< _CharT >, [1892](#), [1893](#)
 - std::ctype< char >, [1904](#), [1905](#)
 - std::ctype< wchar_t >, [1918](#)
 - std::ctype_byname< _CharT >, [1931](#), [1932](#)
 - std::ctype_byname< char >, [1942](#), [1943](#)
- is_always_equal
 - __gnu_cxx::__alloc_traits< _Alloc, typename >, [806](#)
 - std::allocator_traits< _Alloc >, [1094](#)
 - std::allocator_traits< allocator< _Tp > >, [1099](#)
 - std::allocator_traits< allocator< void > >, [1104](#)
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, [1107](#)
- is_bind_expression_v
 - std::experimental, [766](#)
- is_bounded
 - std::__numeric_limits_base, [864](#)
 - std::numeric_limits< _Tp >, [2531](#)
- is_bounded_array_v
 - Variable templates for type traits, [405](#)
- is_constant_evaluated
 - Metaprogramming, [400](#)
- is_corresponding_member
 - Metaprogramming, [400](#)
- is_detected
 - Detection idiom, [314](#)
- is_detected_convertible
 - Detection idiom, [314](#)
- is_detected_convertible_v
 - Detection idiom, [314](#)
- is_detected_exact
 - Detection idiom, [314](#)
- is_detected_exact_v
 - Detection idiom, [314](#)
- is_detected_v
 - Detection idiom, [314](#)
- is_emptyset
 - std::tr2::bool_set, [1776](#)

- is_exact
 - std::__numeric_limits_base, 864
 - std::numeric_limits< _Tp >, 2532
- is_grow_needed
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger<Sorting, 67
External_Load_Access, Size_Type >, 1782
 - __gnu_pbds::sample_resize_trigger, 2725
- is_heap
 - Heap, 92
- is_heap_until
 - Heap, 93
- is_iec559
 - std::__numeric_limits_base, 864
 - std::numeric_limits< _Tp >, 2532
- is_indeterminate
 - std::tr2::bool_set, 1776
- is_integer
 - std::__numeric_limits_base, 864
 - std::numeric_limits< _Tp >, 2532
- is_layout_compatible_v
 - Variable templates for type traits, 405
- is_modulo
 - std::__numeric_limits_base, 864
 - std::numeric_limits< _Tp >, 2532
- is_nothrow_convertible_v
 - Metaprogramming, 401
- is_nothrow_swappable_v
 - Metaprogramming, 401
- is_nothrow_swappable_with_v
 - Metaprogramming, 401
- is_open
 - __gnu_cxx::enc_filebuf< _CharT >, 2026
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2811
 - std::basic_filebuf< _CharT, _Traits >, 1176
 - std::basic_fstream< _CharT, _Traits >, 1210
 - std::basic_ifstream< _CharT, _Traits >, 1261
 - std::basic_ofstream< _CharT, _Traits >, 1445
- is_partitioned
 - Mutating, 27
- is_permutation
 - Non-Mutating, 54–56
- is_placeholder_v
 - std::experimental, 766
- is_pointer_interconvertible_base_of_v
 - Variable templates for type traits, 405
- is_pointer_interconvertible_with_class
 - Metaprogramming, 400
- is_resize_needed
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger<Sorting, 67
External_Load_Access, Size_Type >, 1782
 - __gnu_pbds::sample_resize_policy, 2723
 - __gnu_pbds::sample_resize_trigger, 2725
- is_signed
 - std::__numeric_limits_base, 864
- std::numeric_limits< _Tp >, 2532
- is_singleton
 - std::tr2::bool_set, 1776
- is_sorted
- is_sorted_until
 - Sorting, 67, 68
- is_specialized
 - std::__numeric_limits_base, 865
 - std::numeric_limits< _Tp >, 2532
- is_swappable_v
 - Metaprogramming, 401
- is_swappable_with_v
 - Metaprogramming, 402
- is_unbounded_array_v
 - Variable templates for type traits, 406
- isalnum
 - std, 658
- isalpha
 - std, 658
- isblank
 - std, 658
- iscntrl
 - std, 658
- isctype
 - std::regex_traits< _Ch_type >, 2699
- isdigit
 - std, 658
- isgraph
 - std, 658
- islower
 - std, 659
- isprint
 - std, 659
- ispunct
 - std, 659
- isspace
 - std, 659
- istream, 3474
 - I/O, 171
- istream.tcc, 3219
- istream_iterator
 - std::istream_iterator< _Tp, _CharT, _Traits, _Dist >, 2252
- istream_type
 - std::istreambuf_iterator< _CharT, _Traits >, 2254
- istreambuf_iterator
 - std::istreambuf_iterator< _CharT, _Traits >, 2255
- istream
 - I/O, 171
- isupper
 - std, 659
- isxdigit
 - std, 659

- iter_swap
 - Mutating, [27](#)
- iter_type
 - std::money_get< _CharT, _InIter >, [2398](#)
 - std::money_put< _CharT, _OutIter >, [2402](#)
 - std::num_get< _CharT, _InIter >, [2502](#)
 - std::num_put< _CharT, _OutIter >, [2517](#)
 - std::time_get< _CharT, _InIter >, [2883](#)
 - std::time_put< _CharT, _OutIter >, [2905](#)
- iterator, [3475](#), [3476](#)
 - std::set< _Key, _Compare, _Alloc >, [2747](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3070](#)
- Iterator Tags, [179](#)
- iterator.h, [3515](#)
- iterator.hpp, [3443](#)
- iterator_category
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1731](#)
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1736](#)
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1745](#)
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1748](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, [2265](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, [2269](#)
 - std::back_insert_iterator< _Container >, [1157](#)
 - std::front_insert_iterator< _Container >, [2096](#)
 - std::insert_iterator< _Container >, [2183](#)
 - std::istream_iterator< _Tp, _CharT, _Traits, _Dist >, [2251](#)
 - std::istreambuf_iterator< _CharT, _Traits >, [2254](#)
 - std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >, [2259](#)
 - std::ostream_iterator< _Tp, _CharT, _Traits >, [2569](#)
 - std::ostreambuf_iterator< _CharT, _Traits >, [2573](#)
 - std::raw_storage_iterator< _OutputIterator, _Tp >, [2670](#)
- iterator_concepts.h, [3220](#)
- iterator_fn_imps.hpp, [3428](#)
- Iterators, [173](#)
 - __iterator_category, [177](#)
 - back_inserter, [177](#)
 - front_inserter, [177](#)
 - inserter, [177](#)
 - make_reverse_iterator, [178](#)
 - operator==, [178](#)
- iterators_fn_imps.hpp, [3416](#), [3417](#)
- iword
 - std::basic_fstream< _CharT, _Traits >, [1210](#)
 - std::basic_ifstream< _CharT, _Traits >, [1261](#)
 - std::basic_ios< _CharT, _Traits >, [1295](#)
 - std::basic_iostream< _CharT, _Traits >, [1324](#)
 - std::basic_istream< _CharT, _Traits >, [1369](#)
 - std::basic_istreamstringstream< _CharT, _Traits, _Alloc >, [1408](#)
 - std::basic_ofstream< _CharT, _Traits >, [1445](#)
 - std::basic_ostream< _CharT, _Traits >, [1477](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1509](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1697](#)
 - std::ios_base, [2193](#)
- k
 - std::negative_binomial_distribution< _IntType >, [2487](#)
- key_comp
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2357](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2449](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2474](#)
 - std::set< _Key, _Compare, _Alloc >, [2761](#)
- key_compare
 - std::set< _Key, _Compare, _Alloc >, [2747](#)
- key_eq
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3004](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3033](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3061](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3088](#)
- key_equal
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3043](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3071](#)
- key_type
 - std::set< _Key, _Compare, _Alloc >, [2747](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)

- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3044](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3071](#)
- kill_dependency
 - Atomics, [117](#)
- L1_cache_size
 - __gnu_parallel::Settings, [1076](#)
- L2_cache_size
 - __gnu_parallel::Settings, [1076](#)
- laguerre
 - Mathematical Special Functions, [219](#)
 - TR1 Mathematical Special Functions, [266](#)
- laguerref
 - Mathematical Special Functions, [219](#)
- laguerrel
 - Mathematical Special Functions, [220](#)
- lambda
 - std::exponential_distribution< _RealType >, [2055](#)
- latch, [3476](#)
- launch
 - Futures, [121](#)
- launder
 - std, [659](#)
- lcm
 - std, [660](#)
 - std::experimental, [764](#)
- left
 - std, [660](#)
 - std::basic_fstream< _CharT, _Traits >, [1238](#)
 - std::basic_ifstream< _CharT, _Traits >, [1282](#)
 - std::basic_ios< _CharT, _Traits >, [1305](#)
 - std::basic_iostream< _CharT, _Traits >, [1351](#)
 - std::basic_istream< _CharT, _Traits >, [1388](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1430](#)
 - std::basic_ofstream< _CharT, _Traits >, [1463](#)
 - std::basic_ostream< _CharT, _Traits >, [1495](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1528](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1724](#)
 - std::ios_base, [2200](#)
- left_child_next_sibling_heap.hpp, [3431](#)
- left_child_next_sibling_heap_const_iterator
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, [2266](#)
- left_child_next_sibling_heap_node_point_const_iterator
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, [2270](#)
- legendre
 - Mathematical Special Functions, [220](#)
 - TR1 Mathematical Special Functions, [266](#)
- legendref
 - Mathematical Special Functions, [220](#)
- legendrel
 - Mathematical Special Functions, [221](#)
- length
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [916](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1581](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1638](#)
 - std::match_results< _Bi_iter, _Alloc >, [2372](#)
 - std::regex_traits< _Ch_type >, [2700](#)
 - std::sub_match< _Bilter >, [2854](#)
- lexicographical_compare
 - Sorting, [68](#), [69](#)
- lexicographical_compare_3way
 - SGI, [161](#)
- lfts_config.h, [3396](#)
- Library Fundamentals TS, [309](#)
- limits, [3476](#)
- linear_congruential_engine
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2278](#)
- linear_probe_fn_imp.hpp, [3428](#)
- list, [3478](#), [3479](#)
 - std::list< _Tp, _Alloc >, [2288–2290](#)
- List-Based, [150](#)
- list.tcc, [3221](#)
- list_partition
 - __gnu_parallel, [498](#)
- list_partition.h, [3515](#)
- list_update
 - __gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >, [2307](#)
- list_update_policy.hpp, [3445](#)
- ln10_v
 - std::numbers, [772](#)
- ln2_v
 - std::numbers, [772](#)
- load_factor
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3004](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3033](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3061](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3088](#)
- local_iterator
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3016](#)

- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3044](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3071](#)
- locale, [3479](#)
 - std::locale, [2310–2312](#)
- locale_classes.h, [3222](#)
- locale_classes.tcc, [3222](#)
- locale_conv.h, [3222](#)
- locale_facets.h, [3223](#)
- locale_facets.tcc, [3225](#)
- locale_facets_nonio.h, [3225](#)
- locale_facets_nonio.tcc, [3226](#)
- localefwd.h, [3226](#)
- Locales, [179](#)
 - has_facet, [181](#)
 - use_facet, [181](#)
- lock
 - Mutexes, [123](#)
- log
 - Complex Numbers, [190](#)
- log10
 - Complex Numbers, [190](#)
- log10e_v
 - std::numbers, [773](#)
- log2e_v
 - std::numbers, [773](#)
- logic_error
 - std::logic_error, [2318](#)
- Logical operator traits, [315](#)
- lookup_classname
 - std::regex_traits< _Ch_type >, [2700](#)
- lookup_collatename
 - std::regex_traits< _Ch_type >, [2701](#)
- losertree.h, [3515](#)
- lower_bound
 - Binary Search, [89](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2357–2359](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2449–2451](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2475, 2476](#)
 - std::set< _Key, _Compare, _Alloc >, [2761, 2762](#)
- lowest
 - std::numeric_limits< _Tp >, [2530](#)
- lu_counter_metadata.hpp, [3432](#)
- lu_map.hpp, [3432](#)
- macros.h, [3373](#)
 - _GLIBCXX_DEBUG_VERIFY_COND_AT, [3376](#)
 - _glibcxx_check_erase, [3374](#)
 - _glibcxx_check_erase_after, [3374](#)
 - _glibcxx_check_erase_range, [3374](#)
 - _glibcxx_check_erase_range_after, [3374](#)
 - _glibcxx_check_heap_pred, [3374](#)
 - _glibcxx_check_insert, [3374](#)
 - _glibcxx_check_insert_after, [3374](#)
 - _glibcxx_check_insert_range, [3375](#)
 - _glibcxx_check_insert_range_after, [3375](#)
 - _glibcxx_check_partitioned_lower, [3375](#)
 - _glibcxx_check_partitioned_lower_pred, [3375](#)
 - _glibcxx_check_partitioned_upper_pred, [3375](#)
 - _glibcxx_check_sorted_pred, [3376](#)
- make_any
 - Utilities, [345, 346](#)
- make_array
 - Array creation functions, [311](#)
- make_boyer_moore_horspool_searcher
 - std::experimental, [764](#)
- make_boyer_moore_searcher
 - std::experimental, [765](#)
- make_default_searcher
 - std::experimental, [765](#)
- make_error_code
 - Diagnostics, [131](#)
 - Futures, [121](#)
- make_error_condition
 - Diagnostics, [132](#)
 - Futures, [121](#)
- make_exception_ptr
 - Exceptions, [137](#)
- make_heap
 - Heap, [94](#)
- make_index_sequence
 - std, [633](#)
- make_integer_sequence
 - std, [633](#)
- make_ostream_joiner
 - std::experimental, [765](#)
- make_pair
 - _gnu_parallel::IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [968](#)
 - std::sub_match< _Biliter >, [2855](#)
 - Utilities, [346](#)
- make_reverse_iterator
 - Iterators, [178](#)
- make_shared
 - Pointer Abstractions, [377](#)
- make_signed_t
 - Metaprogramming, [398](#)
- make_unique
 - Pointer Abstractions, [377, 378](#)
- make_unique_for_overwrite
 - Pointer Abstractions, [378, 379](#)
- make_unsigned_t
 - Metaprogramming, [398](#)
- malloc_allocator.h, [3402](#)
- map, [3480](#)

std::map< _Key, _Tp, _Compare, _Alloc >, [2340–2342](#)
 map.h, [3376](#)
 mapped_type
 std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
 std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3017](#)
 mark_count
 std::basic_regex< _Ch_type, _Rx_traits >, [1537](#)
 mask_array
 Numeric Arrays, [228](#)
 mask_array.h, [3227](#)
 mask_based_range_hashing.hpp, [3428](#)
 match_any
 std::regex_constants, [782](#)
 match_continuous
 std::regex_constants, [782](#)
 match_default
 std::regex_constants, [782](#)
 match_flag_type
 std::regex_constants, [776](#)
 match_not_bol
 std::regex_constants, [782](#)
 match_not_bow
 std::regex_constants, [782](#)
 match_not_eol
 std::regex_constants, [782](#)
 match_not_eow
 std::regex_constants, [783](#)
 match_not_null
 std::regex_constants, [783](#)
 match_prev_avail
 std::regex_constants, [783](#)
 match_results
 std::match_results< _Bi_iter, _Alloc >, [2370](#)
 math.h, [3481](#)
 Mathematical constants, [225](#)
 Mathematical Special Functions, [197](#)
 airy_ai, [201](#)
 airy_aif, [201](#)
 airy_ail, [201](#)
 airy_bi, [201](#)
 airy_bif, [201](#)
 airy_bil, [201](#)
 assoc_laguerre, [202](#)
 assoc_laguerref, [202](#)
 assoc_laguerrel, [203](#)
 assoc_legendre, [203](#)
 assoc_legendref, [204](#)
 assoc_legendrel, [204](#)
 beta, [204](#)
 betaf, [205](#)
 betal, [205](#)
 comp_ellint_1, [205](#)
 comp_ellint_1f, [206](#)
 comp_ellint_1l, [206](#)
 comp_ellint_2, [206](#)
 comp_ellint_2f, [207](#)
 comp_ellint_2l, [207](#)
 comp_ellint_3, [207](#)
 comp_ellint_3f, [208](#)
 comp_ellint_3l, [208](#)
 conf_hyperg, [208](#)
 conf_hypergf, [208](#)
 conf_hypergl, [209](#)
 cyl_bessel_i, [209](#)
 cyl_bessel_if, [210](#)
 cyl_bessel_il, [210](#)
 cyl_bessel_j, [210](#)
 cyl_bessel_jf, [210](#)
 cyl_bessel_jl, [211](#)
 cyl_bessel_k, [211](#)
 cyl_bessel_kf, [211](#)
 cyl_bessel_kl, [212](#)
 cyl_neumann, [212](#)
 cyl_neumannf, [212](#)
 cyl_neumannl, [213](#)
 ellint_1, [213](#)
 ellint_1f, [213](#)
 ellint_1l, [214](#)
 ellint_2, [214](#)
 ellint_2f, [215](#)
 ellint_2l, [215](#)
 ellint_3, [215](#)
 ellint_3f, [216](#)
 ellint_3l, [216](#)
 expint, [216](#)
 expintf, [217](#)
 expintl, [217](#)
 hermite, [217](#)
 hermitef, [217](#)
 hermitel, [218](#)
 hyperg, [218](#)
 hypergf, [218](#)
 hypergl, [219](#)
 laguerre, [219](#)
 laguerref, [219](#)
 laguerrel, [220](#)
 legendre, [220](#)
 legendref, [220](#)
 legendrel, [221](#)
 riemann_zeta, [221](#)
 riemann_zetaf, [221](#)
 riemann_zetal, [222](#)
 sph_bessel, [222](#)
 sph_besself, [222](#)
 sph_bessell, [223](#)

- sph_legendre, [223](#)
- sph_legendref, [223](#)
- sph_legendrel, [223](#)
- sph_neumann, [224](#)
- sph_neumannf, [224](#)
- sph_neumannl, [224](#)
- max
 - __gnu_parallel, [498](#)
 - Numeric Arrays, [232](#)
 - Sorting, [69](#), [71](#)
 - std::bernoulli_distribution, [1726](#)
 - std::binomial_distribution< _IntType >, [1755](#)
 - std::cauchy_distribution< _RealType >, [1779](#)
 - std::chi_squared_distribution< _RealType >, [1800](#)
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, [1994](#)
 - std::discrete_distribution< _IntType >, [1997](#)
 - std::exponential_distribution< _RealType >, [2055](#)
 - std::extreme_value_distribution< _RealType >, [2059](#)
 - std::fisher_f_distribution< _RealType >, [2068](#)
 - std::gamma_distribution< _RealType >, [2110](#)
 - std::geometric_distribution< _IntType >, [2113](#)
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, [2174](#)
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2278](#)
 - std::lognormal_distribution< _RealType >, [2324](#)
 - std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, [2384](#)
 - std::negative_binomial_distribution< _IntType >, [2487](#)
 - std::normal_distribution< _RealType >, [2493](#)
 - std::numeric_limits< _Tp >, [2530](#)
 - std::piecewise_constant_distribution< _RealType >, [2621](#)
 - std::piecewise_linear_distribution< _RealType >, [2625](#)
 - std::poisson_distribution< _IntType >, [2635](#)
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, [2787](#)
 - std::student_t_distribution< _RealType >, [2847](#)
 - std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, [2860](#)
 - std::uniform_int_distribution< _IntType >, [2959](#)
 - std::uniform_real_distribution< _RealType >, [2962](#)
 - std::weibull_distribution< _RealType >, [3144](#)
- max_blocks_per_chunk
 - std::pmr::pool_options, [2638](#)
- max_bucket_count
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3004](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3034](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3061](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3088](#)
- max_count
 - __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >, [2327](#)
- max_digits10
 - std::__numeric_limits_base, [865](#)
 - std::numeric_limits< _Tp >, [2532](#)
- max_element
 - Sorting, [71](#), [72](#)
- max_element_minimal_n
 - __gnu_parallel::Settings, [1076](#)
- max_exponent
 - std::__numeric_limits_base, [865](#)
 - std::numeric_limits< _Tp >, [2532](#)
- max_exponent10
 - std::__numeric_limits_base, [865](#)
 - std::numeric_limits< _Tp >, [2533](#)
- max_load_factor
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3004](#), [3005](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3034](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3061](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3088](#)
- max_size
 - __gnu_cxx::__alloc_traits< _Alloc, typename >, [809](#)
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [916](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1581](#)
 - std::allocator_traits< _Alloc >, [1097](#)
 - std::allocator_traits< allocator< _Tp > >, [1102](#)
 - std::allocator_traits< allocator< void > >, [1105](#)
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, [1110](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1638](#)
 - std::deque< _Tp, _Alloc >, [1982](#)
 - std::forward_list< _Tp, _Alloc >, [2085](#)
 - std::list< _Tp, _Alloc >, [2297](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2359](#)
 - std::match_results< _Bi_iter, _Alloc >, [2373](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2451](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2476](#)
 - std::set< _Key, _Compare, _Alloc >, [2763](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2011](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3005](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3034](#)

- [_Alloc >, 3034](#)
 - [std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3061](#)
 - [std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3089](#)
 - [std::vector< _Tp, _Alloc >, 3116](#)
- [max_size_type.h, 3227](#)
- [mean](#)
 - [std::normal_distribution< _RealType >, 2493](#)
 - [std::poisson_distribution< _IntType >, 2635](#)
- [mem_fn](#)
 - [Function Objects, 352](#)
- [Memory, 361](#)
 - [align, 362](#)
 - [assume_aligned, 362](#)
 - [uninitialized_copy, 363](#)
 - [uninitialized_copy_n, 363](#)
 - [uninitialized_default_construct, 363](#)
 - [uninitialized_default_construct_n, 364](#)
 - [uninitialized_fill, 364](#)
 - [uninitialized_fill_n, 365](#)
 - [uninitialized_move, 365](#)
 - [uninitialized_move_n, 365](#)
 - [uninitialized_value_construct, 366](#)
 - [uninitialized_value_construct_n, 366](#)
- [memory, 3481–3483](#)
- [memory_order](#)
 - [Atomics, 117](#)
- [memory_resource, 3483, 3484](#)
 - [get_default_resource, 3485](#)
 - [new_delete_resource, 3485](#)
 - [null_memory_resource, 3485](#)
 - [set_default_resource, 3485](#)
- [memoryfwd.h, 3227](#)
- [merge](#)
 - [Sorting, 72, 73](#)
 - [std::forward_list< _Tp, _Alloc >, 2085](#)
 - [std::list< _Tp, _Alloc >, 2297, 2299](#)
- [merge.h, 3516](#)
- [merge_minimal_n](#)
 - [__gnu_parallel:: Settings, 1076](#)
- [merge_oversampling](#)
 - [__gnu_parallel:: Settings, 1076](#)
- [mersenne_twister_engine](#)
 - [std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, 2384](#)
- [message](#)
 - [std::error_code, 2049](#)
 - [std::error_condition, 2050](#)
- [messages](#)
 - [std::locale, 2316](#)
 - [std::messages< _CharT >, 2388](#)
- [messages_members.h, 3593](#)
- [metadata_const_reference](#)
 - [__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >, 1731](#)
 - [__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >, 1736](#)
 - [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1002](#)
 - [__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1008](#)
- [metadata_reference](#)
 - [__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >, 2327](#)
 - [__gnu_pbds::lu_move_to_front_policy< _Alloc >, 2331](#)
- [metadata_type](#)
 - [__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >, 1731](#)
 - [__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >, 1736](#)
 - [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1002](#)
 - [__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1008](#)
 - [__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >, 2327](#)
 - [__gnu_pbds::lu_move_to_front_policy< _Alloc >, 2331](#)
 - [__gnu_pbds::sample_update_policy, 2731](#)
- [Metaprogramming, 388](#)
 - [__cpp_lib_remove_cvref, 396](#)
 - [__cpp_lib_type_identity, 396](#)
 - [add_const_t, 397](#)
 - [add_cv_t, 397](#)
 - [add_lvalue_reference_t, 397](#)
 - [add_pointer_t, 397](#)
 - [add_rvalue_reference_t, 397](#)
 - [add_volatile_t, 397](#)
 - [aligned_storage_t, 397](#)
 - [alignment_value, 401](#)
 - [bool_constant, 397](#)
 - [common_type_t, 397](#)
 - [conditional_t, 398](#)
 - [decay_t, 398](#)
 - [enable_if_t, 398](#)
 - [false_type, 398](#)
 - [invoke_result_t, 398](#)
 - [is_constant_evaluated, 400](#)
 - [is_corresponding_member, 400](#)
 - [is_nothrow_convertible_v, 401](#)
 - [is_nothrow_swappable_v, 401](#)

- is_nothrow_swappable_with_v, 401
- is_pointer_interconvertible_with_class, 400
- is_swappable_v, 401
- is_swappable_with_v, 402
- make_signed_t, 398
- make_unsigned_t, 398
- remove_all_extents_t, 398
- remove_const_t, 398
- remove_cv_t, 399
- remove_cvref_t, 399
- remove_extent_t, 399
- remove_pointer_t, 399
- remove_reference_t, 399
- remove_volatile_t, 399
- result_of_t, 399
- swap, 400, 401
- true_type, 399
- type_identity_t, 399
- underlying_type_t, 400
- void_t, 400
- microseconds
 - Time, 413
- milliseconds
 - Time, 413
- min
 - __gnu_parallel, 499
 - Numeric Arrays, 232
 - Sorting, 73, 74
 - std::bernoulli_distribution, 1727
 - std::binomial_distribution< _IntType >, 1755
 - std::cauchy_distribution< _RealType >, 1779
 - std::chi_squared_distribution< _RealType >, 1800
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1994
 - std::discrete_distribution< _IntType >, 1997
 - std::exponential_distribution< _RealType >, 2055
 - std::extreme_value_distribution< _RealType >, 2059
 - std::fisher_f_distribution< _RealType >, 2068
 - std::gamma_distribution< _RealType >, 2110
 - std::geometric_distribution< _IntType >, 2113
 - std::independent_bits_engine< __RandomNumberEngine, __w, _UIntType >, 2174
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2278
 - std::lognormal_distribution< _RealType >, 2324
 - std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, 2384
 - std::negative_binomial_distribution< _IntType >, 2487
 - std::normal_distribution< _RealType >, 2493
 - std::numeric_limits< _Tp >, 2530
 - std::piecewise_constant_distribution< _RealType >, 2621
 - std::piecewise_linear_distribution< _RealType >, 2625
 - std::poisson_distribution< _IntType >, 2635
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, 2787
 - std::student_t_distribution< _RealType >, 2847
 - std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2860
 - std::uniform_int_distribution< _IntType >, 2959
 - std::uniform_real_distribution< _RealType >, 2962
 - std::weibull_distribution< _RealType >, 3144
- min_element
 - Sorting, 75
- min_element_minimal_n
 - __gnu_parallel::Settings, 1076
- min_exponent
 - std::__numeric_limits_base, 865
 - std::numeric_limits< _Tp >, 2533
- min_exponent10
 - std::__numeric_limits_base, 865
 - std::numeric_limits< _Tp >, 2533
- minmax
 - Sorting, 76
- minmax_element
 - Sorting, 76, 77
- minstd_rand
 - Random Number Generators, 258
- minstd_rand0
 - Random Number Generators, 258
- minutes
 - Time, 413
- mismatch
 - Non-Mutating, 56–58
- mod_based_range_hashing.hpp, 3428
- modulus
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2281
- mofunc_impl.h, 3227
- monetary
 - std::locale, 2316
- money_get
 - std::money_get< _CharT, _InIter >, 2398
- money_put
 - std::money_put< _CharT, _OutIter >, 2402
- money_punct
 - std::money_punct< _CharT, _Intl >, 2408
- months
 - Time, 413
- move
 - Mutating, 27
 - Utilities, 346
- move.h, 3228
- move_backward
 - Mutating, 28

- move_if_noexcept
 - Utilities, [347](#)
- move_only_function
 - std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>, [2424](#), [2425](#)
- move_only_function.h, [3229](#)
- mt19937
 - Random Number Generators, [258](#)
- mt19937_64
 - Random Number Generators, [258](#)
- mt_allocator.h, [3402](#)
- multiline
 - std::regex_constants, [783](#)
- multimap
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2434](#)–[2436](#)
- multimap.h, [3377](#)
- multiplier
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2281](#)
- multiseq_partition
 - __gnu_parallel, [499](#)
- multiseq_selection
 - __gnu_parallel, [499](#)
- multiseq_selection.h, [3516](#)
- multiset
 - std::multiset< _Key, _Compare, _Alloc >, [2462](#)–[2464](#)
- multiset.h, [3378](#)
- multiway_merge
 - __gnu_parallel, [500](#)
- multiway_merge.h, [3517](#)
 - _GLIBCXX_PARALLEL_LENGTH, [3520](#)
- multiway_merge_3_variant
 - __gnu_parallel, [501](#)
- multiway_merge_4_variant
 - __gnu_parallel, [502](#)
- multiway_merge_exact_splitting
 - __gnu_parallel, [502](#)
- multiway_merge_loser_tree
 - __gnu_parallel, [503](#)
- multiway_merge_loser_tree_sentinel
 - __gnu_parallel, [503](#)
- multiway_merge_loser_tree_unguarded
 - __gnu_parallel, [504](#)
- multiway_merge_minimal_k
 - __gnu_parallel:: Settings, [1076](#)
- multiway_merge_minimal_n
 - __gnu_parallel:: Settings, [1076](#)
- multiway_merge_oversampling
 - __gnu_parallel:: Settings, [1076](#)
- multiway_merge_sampling_splitting
 - __gnu_parallel, [504](#)
- multiway_merge_sentinels
 - __gnu_parallel, [505](#)
- multiway_mergesort.h, [3520](#)
- Mutating, [20](#)
 - copy, [22](#)
 - copy_backward, [23](#)
 - copy_if, [23](#)
 - copy_n, [24](#)
 - fill, [25](#)
 - fill_n, [25](#)
 - generate, [26](#)
 - generate_n, [26](#)
 - is_partitioned, [27](#)
 - iter_swap, [27](#)
 - move, [27](#)
 - move_backward, [28](#)
 - partition, [29](#)
 - partition_copy, [29](#)
 - partition_point, [30](#)
 - random_shuffle, [30](#), [31](#)
 - remove, [31](#)
 - remove_copy, [32](#)
 - remove_copy_if, [32](#)
 - remove_if, [33](#)
 - replace, [33](#)
 - replace_copy_if, [34](#)
 - replace_if, [35](#)
 - reverse, [35](#)
 - reverse_copy, [36](#)
 - rotate, [36](#)
 - rotate_copy, [37](#)
 - shuffle, [37](#)
 - stable_partition, [38](#)
 - swap_ranges, [38](#)
 - transform, [39](#)
 - unique, [40](#), [41](#)
 - unique_copy, [41](#), [42](#)
- mutex, [3485](#)
- Mutexes, [122](#)
 - adopt_lock, [125](#)
 - call_once, [123](#)
 - defer_lock, [125](#)
 - lock, [123](#)
 - swap, [124](#)
 - try_lock, [124](#)
 - try_to_lock, [125](#)
- name
 - std::V2::error_category, [2047](#)
 - std::locale, [2313](#)
 - std::type_info, [2951](#)
- nanoseconds
 - Time, [413](#)
- narrow

- std::ctype_abstract_base< _CharT >, [832](#), [833](#)
- std::basic_fstream< _CharT, _Traits >, [1211](#)
- std::basic_ifstream< _CharT, _Traits >, [1261](#)
- std::basic_ios< _CharT, _Traits >, [1296](#)
- std::basic_iostream< _CharT, _Traits >, [1324](#)
- std::basic_istream< _CharT, _Traits >, [1369](#)
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1408](#)
- std::basic_ofstream< _CharT, _Traits >, [1445](#)
- std::basic_ostream< _CharT, _Traits >, [1478](#)
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1510](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1697](#)
- std::ctype< _CharT >, [1893](#), [1894](#)
- std::ctype< char >, [1905](#)
- std::ctype< wchar_t >, [1919](#)
- std::ctype_byname< _CharT >, [1932](#), [1933](#)
- std::ctype_byname< char >, [1943](#)
- native_handle
 - std::thread, [2873](#)
- neg_format
 - std::moneypunct< _CharT, _Intl >, [2412](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2420](#)
- negative_sign
 - std::moneypunct< _CharT, _Intl >, [2413](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2420](#)
- Negators, [360](#)
 - not1, [360](#)
 - not2, [360](#)
- nested_exception.h, [3229](#)
- new, [3153](#)
 - operator new, [3154](#)
- new_allocator.h, [3229](#)
- new_delete_resource
 - memory_resource, [3485](#)
- new_handler
 - std, [633](#)
- next_permutation
 - Sorting, [77](#), [79](#)
- noboolalpha
 - std, [660](#)
- Node handles, [128](#)
- node.hpp, [3431](#), [3432](#)
- node_begin
 - __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2578](#)
 - __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >, [2612](#)
 - __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2679](#)
 - __gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2799](#)
- node_const_iterator
 - __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >, [1738](#)
 - __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >, [1739](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [2919](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [2920](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [2922](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [2922](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [2923](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [2925](#)
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >, [2939](#)
 - __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >, [2940](#)
- node_end
 - __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2578](#)
 - __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >, [2612](#), [2613](#)
 - __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2679](#)
 - __gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >, [2799](#)
- node_handle.h, [3230](#)
- node_iterators.hpp, [3404](#)
- node_metadata_selector.hpp, [3440](#)
- node_type
 - __gnu_pbds::detail::pat_trie_base, [2610](#)
 - __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >, [2612](#)
- node_update
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >, [2939](#)
 - __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >, [2940](#)
- Non-Mutating, [43](#)

- adjacent_find, [45](#)
- all_of, [45](#)
- any_of, [46](#)
- count, [46](#)
- count_if, [47](#)
- equal, [47–49](#)
- find, [49](#)
- find_end, [50, 51](#)
- find_first_of, [51, 52](#)
- find_if, [52](#)
- find_if_not, [53](#)
- for_each, [53](#)
- for_each_n, [54](#)
- is_permutation, [54–56](#)
- mismatch, [56–58](#)
- none_of, [59](#)
- search, [59, 60](#)
- search_n, [60, 61](#)
- none
 - std::bitset< _Nb >, [1770](#)
 - std::locale, [2316](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2011](#)
- none_of
 - Non-Mutating, [59](#)
- norm
 - Complex Numbers, [190](#)
- Normal Distributions, [246](#)
 - operator!=, [247, 248](#)
 - operator<<, [248](#)
 - operator>>, [248](#)
- normal_distribution
 - std::normal_distribution< _RealType >, [2493](#)
- noshowbase
 - std, [660](#)
- noshowpoint
 - std, [660](#)
- noshowpos
 - std, [660](#)
- noskipws
 - std, [660](#)
- nosubs
 - std::regex_constants, [783](#)
- not1
 - Negators, [360](#)
- not2
 - Negators, [360](#)
- not_fn
 - Function Objects, [352](#)
 - std::experimental, [765](#)
- notify_cleared
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1782](#)
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2157](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2725](#)
- notify_erase_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1782](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_erase_search_end
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1782](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_erase_search_start
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1782](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_erased
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_externally_resized
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_find_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_find_search_end
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_find_search_start
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2723](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_insert_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_insert_search_end
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1783](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
- notify_insert_search_start
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1784](#)

- __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2726](#)
- notify_inserted
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1784](#)
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2157](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2727](#)
- notify_resized
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1784](#)
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2157](#)
 - __gnu_pbds::sample_range_hashing, [2720](#)
 - __gnu_pbds::sample_ranged_hash_fn, [2721](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2727](#)
- nounitbuf
 - std, [660](#)
- nouppercase
 - std, [661](#)
- npos
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [933](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1590](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1656](#)
- nth_element
 - Sorting, [79](#), [80](#)
- nth_element_minimal_n
 - __gnu_parallel::Settings, [1077](#)
- null_memory_resource
 - memory_resource, [3485](#)
- null_node_metadata.hpp, [3421](#)
- nullopt
 - Optional values, [316](#)
 - Utilities, [350](#)
- num_blocks
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2011](#)
- num_children
 - __gnu_pbds::detail::pat_trie_base::Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >, [1002](#)
 - __gnu_pbds::detail::pat_trie_base::Node_iter< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >, [1008](#)
- num_get
 - std::num_get< _CharT, _InIter >, [2502](#)
- num_put
 - std::num_put< _CharT, _OutIter >, [2518](#)
- numbers, [3486](#)
- numeric, [3487–3489](#)
 - std::locale, [2316](#)
- Numeric Arrays, [225](#)
 - ~gslice, [230](#)
 - apply, [230](#), [231](#)
 - begin, [231](#)
 - cshift, [231](#)
 - end, [232](#)
 - gslice, [227](#), [228](#)
 - gslice_array, [228](#)
 - indirect_array, [228](#)
 - mask_array, [228](#)
 - min, [232](#)
 - operator=, [233–236](#)
 - operator[], [236–239](#)
 - resize, [240](#)
 - shift, [240](#)
 - size, [240](#), [241](#)
 - slice, [228](#)
 - slice_array, [229](#)
 - start, [241](#)
 - stride, [241](#)
 - sum, [241](#)
 - swap, [241](#)
 - valarray, [229](#), [230](#)
- numeric_traits.h, [3396](#)
- numeric_fwd.h, [3520](#)
- Numerics, [182](#)
- num_punct
 - std::num_punct< _CharT >, [2553](#), [2554](#)
- oct
 - std, [661](#)
 - std::basic_fstream< _CharT, _Traits >, [1238](#)
 - std::basic_ifstream< _CharT, _Traits >, [1282](#)
 - std::basic_ios< _CharT, _Traits >, [1305](#)
 - std::basic_iostream< _CharT, _Traits >, [1351](#)
 - std::basic_istream< _CharT, _Traits >, [1388](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1430](#)
 - std::basic_ofstream< _CharT, _Traits >, [1464](#)
 - std::basic_ostream< _CharT, _Traits >, [1496](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1528](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1724](#)
 - std::ios_base, [2200](#)
- off_type
 - std::basic_ios< _CharT, _Traits >, [1289](#)
 - std::basic_streambuf< _CharT, _Traits >, [1542](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3128](#)
- ofstream
 - I/O, [171](#)
- omp_loop.h, [3522](#)
- omp_loop_static.h, [3522](#)

- once_flag
 - std::once_flag, [2563](#)
- open
 - __gnu_cxx::enc_filebuf< _CharT >, [2026](#), [2027](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2811](#), [2812](#)
 - std::basic_filebuf< _CharT, _Traits >, [1177](#), [1178](#)
 - std::basic_fstream< _CharT, _Traits >, [1211](#), [1212](#)
 - std::basic_ifstream< _CharT, _Traits >, [1261](#), [1262](#)
 - std::basic_ofstream< _CharT, _Traits >, [1446](#)
- openmode
 - std::basic_fstream< _CharT, _Traits >, [1198](#)
 - std::basic_ifstream< _CharT, _Traits >, [1249](#)
 - std::basic_ios< _CharT, _Traits >, [1289](#)
 - std::basic_iostream< _CharT, _Traits >, [1313](#)
 - std::basic_istream< _CharT, _Traits >, [1358](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1396](#)
 - std::basic_ofstream< _CharT, _Traits >, [1438](#)
 - std::basic_ostream< _CharT, _Traits >, [1471](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1503](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1686](#)
 - std::ios_base, [2191](#)
- operator __sv_type
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1581](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1638](#)
- operator _Iterator
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, [1037](#)
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, [1049](#)
- operator _RAIter
 - __gnu_parallel::GuardedIterator< _RAIter, _Compare >, [960](#)
- operator bool
 - std::basic_fstream< _CharT, _Traits >, [1212](#)
 - std::basic_ifstream< _CharT, _Traits >, [1262](#)
 - std::basic_ios< _CharT, _Traits >, [1296](#)
 - std::basic_iostream< _CharT, _Traits >, [1325](#)
 - std::basic_istream< _CharT, _Traits >, [1370](#)
 - std::basic_istream< _CharT, _Traits >::sentry, [2738](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1409](#)
 - std::basic_ofstream< _CharT, _Traits >, [1447](#)
 - std::basic_ostream< _CharT, _Traits >, [1478](#)
 - std::basic_ostream< _CharT, _Traits >::sentry, [2739](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1510](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1698](#)
 - std::error_code, [2049](#)
 - std::error_condition, [2050](#)
 - std::function< _Res(_ArgTypes...)>, [2100](#)
 - std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>, [2425](#)
 - std::shared_ptr< _Tp >, [2783](#)
 - std::tr2::bool_set, [1776](#)
 - std::unique_ptr< _Tp, _Dp >, [2968](#)
 - std::unique_ptr< _Tp[], _Dp >, [2973](#)
- operator new
 - new, [3154](#)
- operator streamoff
 - std::fpos< _StateT >, [2092](#)
- operator string_type
 - std::sub_match< _Biter >, [2854](#)
- operator!
 - std::basic_fstream< _CharT, _Traits >, [1212](#)
 - std::basic_ifstream< _CharT, _Traits >, [1263](#)
 - std::basic_ios< _CharT, _Traits >, [1296](#)
 - std::basic_iostream< _CharT, _Traits >, [1325](#)
 - std::basic_istream< _CharT, _Traits >, [1370](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1409](#)
 - std::basic_ofstream< _CharT, _Traits >, [1447](#)
 - std::basic_ostream< _CharT, _Traits >, [1478](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1510](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1698](#)
 - std::valarray< _Tp >, [3096](#)
- operator!=
 - __gnu_cxx, [440](#), [441](#)
 - __gnu_parallel::IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [968](#)
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1732](#)
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, [1737](#)
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1746](#)
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, [1749](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, [2266](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, [2270](#)
 - __gnu_pbds::detail::pat_trie_base::Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, [1003](#)
 - __gnu_pbds::detail::pat_trie_base::Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, [1008](#)
 - Bernoulli Distributions, [244](#)
 - Dynamic Bitset., [142](#)
 - Normal Distributions, [247](#), [248](#)

- Pointer Abstractions, [379](#), [380](#)
- Poisson Distributions, [250](#), [251](#)
- Random Number Generators, [258–260](#)
- Regular Expressions, [273–276](#)
- std, [661–663](#)
- std::filesystem::path, [2619](#)
- std::rel_ops, [784](#)
- std::sub_match<_Biliter >, [2856](#)
- Time, [417](#)
- Uniform Distributions, [254](#)
- Utilities, [347](#)
- operator<
 - __gnu_cxx, [443](#), [444](#)
 - __gnu_parallel::GuardedIterator<_RAIter, _Compare >, [961](#)
 - __gnu_parallel::IteratorPair<_Iterator1, _Iterator2, _IteratorCategory >, [968](#)
- Diagnostics, [132](#)
- Pointer Abstractions, [380](#), [381](#)
- Regular Expressions, [276–278](#)
- std, [665–670](#)
- std::_V2::error_category, [2047](#)
- std::filesystem::path, [2619](#)
- std::sub_match<_Biliter >, [2856](#)
- Time, [421](#)
- Utilities, [347](#)
- operator<<
 - Bernoulli Distributions, [244](#), [245](#)
 - Complex Numbers, [193](#)
 - Diagnostics, [132](#)
 - Dynamic Bitset., [143](#)
 - Normal Distributions, [248](#)
 - Pointer Abstractions, [381](#)
 - Poisson Distributions, [251](#), [252](#)
 - Random Number Generators, [260](#)
 - Regular Expressions, [278](#)
 - std, [671–681](#)
 - std::__detail, [723](#)
 - std::basic_fstream<_CharT, _Traits >, [1212–1217](#)
 - std::basic_istream<_CharT, _Traits >, [1325–1330](#)
 - std::basic_ofstream<_CharT, _Traits >, [1447–1452](#)
 - std::basic_ostream<_CharT, _Traits >, [1479](#), [1480](#), [1482–1485](#)
 - std::basic_ostringstream<_CharT, _Traits, _Alloc >, [1510–1516](#)
 - std::basic_stringstream<_CharT, _Traits, _Alloc >, [1698–1703](#)
 - std::binomial_distribution<_IntType >, [1756](#)
 - std::bitset<_Nb >, [1771](#)
 - std::chi_squared_distribution<_RealType >, [1801](#)
 - std::discard_block_engine<_RandomNumberEngine, __p, __r >, [1995](#)
 - std::discrete_distribution<_IntType >, [1998](#)
 - std::filesystem::path, [2619](#)
 - std::fisher_f_distribution<_RealType >, [2069](#)
 - std::gamma_distribution<_RealType >, [2111](#)
 - std::linear_congruential_engine<_UIntType, __a, __c, __m >, [2280](#)
 - std::lognormal_distribution<_RealType >, [2325](#)
 - std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, [2384](#)
 - std::negative_binomial_distribution<_IntType >, [2488](#)
 - std::normal_distribution<_RealType >, [2494](#)
 - std::piecewise_constant_distribution<_RealType >, [2622](#)
 - std::piecewise_linear_distribution<_RealType >, [2626](#)
 - std::poisson_distribution<_IntType >, [2636](#)
 - std::shuffle_order_engine<_RandomNumberEngine, __k >, [2788](#)
 - std::student_t_distribution<_RealType >, [2848](#)
 - std::subtract_with_carry_engine<_UIntType, __w, __s, __r >, [2861](#)
 - std::tr2::dynamic_bitset<_WordT, _Alloc >, [2012](#)
 - Uniform Distributions, [255](#)
- operator<=<=
 - std::bitset<_Nb >, [1771](#)
 - std::gslice_array<_Tp >, [2132](#)
 - std::indirect_array<_Tp >, [2178](#)
 - std::mask_array<_Tp >, [2366](#)
 - std::slice_array<_Tp >, [2793](#)
 - std::tr2::dynamic_bitset<_WordT, _Alloc >, [2012](#)
 - std::valarray<_Tp >, [3098](#)
- operator<=
 - __gnu_cxx, [444](#), [445](#)
 - __gnu_parallel::GuardedIterator<_RAIter, _Compare >, [961](#)
 - __gnu_parallel::IteratorPair<_Iterator1, _Iterator2, _IteratorCategory >, [968](#)
- Dynamic Bitset., [143](#)
- Pointer Abstractions, [382](#)
- Regular Expressions, [279–281](#)
- std, [681–683](#)
- std::__debug, [719](#)
- std::filesystem::path, [2619](#)
- std::rel_ops, [784](#)
- std::sub_match<_Biliter >, [2856](#)
- Time, [421](#)
- Utilities, [348](#)
- operator>
 - __gnu_cxx, [446](#), [447](#)
 - __gnu_parallel::IteratorPair<_Iterator1, _Iterator2, _IteratorCategory >, [968](#)
- Dynamic Bitset., [143](#)
- Pointer Abstractions, [383](#), [384](#)
- Regular Expressions, [284–286](#)

- std, 689–691
- std::__debug, 719
- std::filesystem::path, 2619
- std::rel_ops, 785
- std::sub_match< _Bilter >, 2856
- Time, 421
- Utilities, 348
- operator>>
 - Bernoulli Distributions, 245, 246
 - Complex Numbers, 194
 - Dynamic Bitset., 143
 - Normal Distributions, 248
 - Poisson Distributions, 252, 253
 - std, 694–698
 - std::__detail, 724
 - std::basic_fstream< _CharT, _Traits >, 1218–1220, 1222–1224
 - std::basic_ifstream< _CharT, _Traits >, 1263–1265, 1267–1269
 - std::basic_istream< _CharT, _Traits >, 1330–1336
 - std::basic_istream< _CharT, _Traits >, 1370–1375
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1409–1414
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1703–1709
 - std::binomial_distribution< _IntType >, 1757
 - std::bitset< _Nb >, 1771
 - std::chi_squared_distribution< _RealType >, 1801
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1996
 - std::discrete_distribution< _IntType >, 1999
 - std::filesystem::path, 2620
 - std::fisher_f_distribution< _RealType >, 2069
 - std::gamma_distribution< _RealType >, 2112
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2175
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2280
 - std::lognormal_distribution< _RealType >, 2325
 - std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, 2385
 - std::negative_binomial_distribution< _IntType >, 2488
 - std::normal_distribution< _RealType >, 2495
 - std::piecewise_constant_distribution< _RealType >, 2623
 - std::piecewise_linear_distribution< _RealType >, 2626
 - std::poisson_distribution< _IntType >, 2637
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, 2788
 - std::student_t_distribution< _RealType >, 2849
 - std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2862
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2012
 - Uniform Distributions, 255, 256
- operator>=
 - std::bitset< _Nb >, 1771
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2178
 - std::mask_array< _Tp >, 2366
 - std::slice_array< _Tp >, 2793
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2012
 - std::valarray< _Tp >, 3098
- operator>=
 - __gnu_cxx, 448
 - __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, 969
 - Dynamic Bitset., 143
 - Pointer Abstractions, 384, 385
 - Regular Expressions, 286–288
 - std, 691–693
 - std::__debug, 719
 - std::filesystem::path, 2620
 - std::rel_ops, 785
 - std::sub_match< _Bilter >, 2856
 - Time, 421
 - Utilities, 348
- operator*
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1037
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1049
 - __gnu_parallel::GuardedIterator< _RAIter, _Compare >, 960
 - __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1732
 - __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1737
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1746
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1749
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, 2267
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, 2270
 - __gnu_pbds::detail::ov_tree_node_it< Value_Type, Metadata_Type, _Alloc >, 2582
 - __gnu_pbds::detail::pat_trie_base::Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1003
 - __gnu_pbds::detail::pat_trie_base::Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >, 1009
 - Complex Numbers, 191
 - std::auto_ptr< _Tp >, 1154

- std::back_insert_iterator< _Container >, 1157
- std::front_insert_iterator< _Container >, 2096
- std::insert_iterator< _Container >, 2183
- std::istreambuf_iterator< _CharT, _Traits >, 2255
- std::ostreambuf_iterator< _CharT, _Traits >, 2574
- std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2693
- std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2697
- std::reverse_iterator< _Iterator >, 2710
- std::unique_ptr< _Tp, _Dp >, 2968
- Time, 419
- operator*=
 - Complex Numbers, 191
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2177
 - std::mask_array< _Tp >, 2366
 - std::slice_array< _Tp >, 2792
 - std::valarray< _Tp >, 3097
- operator~
 - std::bitset< _Nb >, 1773
 - std::regex_constants, 780
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2014
 - std::valarray< _Tp >, 3099
- operator^
 - Dynamic Bitset., 144
 - std, 698
 - std::regex_constants, 778
- operator^=
 - std::bitset< _Nb >, 1772
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2178
 - std::mask_array< _Tp >, 2366
 - std::regex_constants, 779
 - std::slice_array< _Tp >, 2793
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2014
 - std::valarray< _Tp >, 3098
- operator()
 - __gnu_cxx::subtractive_rng, 2864
 - __gnu_parallel::__Nothing, 1011
 - __gnu_parallel::__RandomNumber, 1023
 - __gnu_parallel::__accumulate_selector< _It >, 801
 - __gnu_parallel::__adjacent_find_selector, 804
 - __gnu_parallel::__count_if_selector< _It, _Diff >, 822
 - __gnu_parallel::__count_selector< _It, _Diff >, 823
 - __gnu_parallel::__fill_selector< _It >, 840
 - __gnu_parallel::__find_first_of_selector< _FIterator >, 842
 - __gnu_parallel::__find_if_selector, 843
 - __gnu_parallel::__for_each_selector< _It >, 845
 - __gnu_parallel::__generate_selector< _It >, 846
 - __gnu_parallel::__identity_selector< _It >, 850
 - __gnu_parallel::__inner_product_selector< _It, _It2, _Tp >, 851
 - __gnu_parallel::__mismatch_selector, 856
 - __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >, 875
 - __gnu_parallel::__replace_selector< _It, _Tp >, 877
 - __gnu_parallel::__transform1_selector< _It >, 878
 - __gnu_parallel::__transform2_selector< _It >, 879
 - __gnu_pbds::direct_mask_range_hashing< Size_Type >, 1988
 - __gnu_pbds::direct_mod_range_hashing< Size_Type >, 1989
 - __gnu_pbds::linear_probe_fn< Size_Type >, 2282
 - __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >, 2327
 - __gnu_pbds::lu_move_to_front_policy< _Alloc >, 2331
 - __gnu_pbds::quadratic_probe_fn< Size_Type >, 2650
 - __gnu_pbds::sample_probe_fn, 2719
 - __gnu_pbds::sample_range_hashing, 2720
 - __gnu_pbds::sample_ranged_hash_fn, 2721
 - __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2730
 - __gnu_pbds::sample_update_policy, 2731
 - __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >, 2917
 - __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2930
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, 2934
 - std::bernoulli_distribution, 1727
 - std::binomial_distribution< _IntType >, 1755
 - std::cauchy_distribution< _RealType >, 1779
 - std::chi_squared_distribution< _RealType >, 1800
 - std::default_delete< _Tp >, 1955
 - std::default_delete< _Tp[] >, 1956
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1994
 - std::discrete_distribution< _IntType >, 1998
 - std::exponential_distribution< _RealType >, 2055
 - std::extreme_value_distribution< _RealType >, 2059
 - std::fisher_f_distribution< _RealType >, 2068
 - std::function< _Res(_ArgTypes...) >, 2100
 - std::gamma_distribution< _RealType >, 2110, 2111
 - std::geometric_distribution< _IntType >, 2114
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2174
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2279
 - std::locale, 2313
 - std::lognormal_distribution< _RealType >, 2324
 - std::move_only_function< _Res(_ArgTypes...) >, 2425
 - std::move_only_function< _Res(_ArgTypes...) >, 2425

- std::negative_binomial_distribution< _IntType >, 2487
- std::normal_distribution< _RealType >, 2493
- std::piecewise_constant_distribution< _RealType >, 2622
- std::piecewise_linear_distribution< _RealType >, 2625
- std::plus< _Tp >, 2628
- std::poisson_distribution< _IntType >, 2635, 2636
- std::shuffle_order_engine< _RandomNumberEngine, __k >, 2787
- std::student_t_distribution< _RealType >, 2847
- std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2860
- std::uniform_int_distribution< _IntType >, 2959
- std::uniform_real_distribution< _RealType >, 2962
- std::weibull_distribution< _RealType >, 3144
- operator+
 - __gnu_cxx, 441–443
 - Complex Numbers, 191, 192
 - std, 664, 665
 - std::fpos< _StateT >, 2092
 - std::reverse_iterator< _Iterator >, 2710
 - std::valarray< _Tp >, 3097
 - Time, 419, 420
- operator++
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1037
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1049, 1050
 - __gnu_parallel::GuardedIterator< _RAIter, _Compare >, 960
 - std::back_insert_iterator< _Container >, 1158
 - std::front_insert_iterator< _Container >, 2096
 - std::insert_iterator< _Container >, 2183
 - std::istreambuf_iterator< _CharT, _Traits >, 2255
 - std::ostreambuf_iterator< _CharT, _Traits >, 2574
 - std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2693
 - std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2697
 - std::reverse_iterator< _Iterator >, 2710
- operator+=
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 917, 918
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1581
 - Complex Numbers, 192
 - std::basic_string< _CharT, _Traits, _Alloc >, 1638, 1639
 - std::complex< _Tp >, 1853
 - std::fpos< _StateT >, 2092
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2177
- std::mask_array< _Tp >, 2366
- std::reverse_iterator< _Iterator >, 2711
- std::slice_array< _Tp >, 2792
- std::valarray< _Tp >, 3097
- operator-
 - Complex Numbers, 192
 - Dynamic Bitset., 142
 - std::fpos< _StateT >, 2092
 - std::reverse_iterator< _Iterator >, 2711
 - std::valarray< _Tp >, 3097
 - Time, 420
- operator->
 - __gnu_debug::Safe_iterator< _Iterator, _Sequence, _Category >, 1037
 - __gnu_debug::Safe_local_iterator< _Iterator, _Sequence >, 1050
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1746
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1749
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, 2267
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, 2270
 - std::auto_ptr< _Tp >, 1154
 - std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2693
 - std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2697
 - std::reverse_iterator< _Iterator >, 2712
 - std::unique_ptr< _Tp, _Dp >, 2968
- operator--
 - std::reverse_iterator< _Iterator >, 2711
- operator-=
 - Complex Numbers, 193
 - std::complex< _Tp >, 1853
 - std::fpos< _StateT >, 2092
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2178
 - std::mask_array< _Tp >, 2366
 - std::reverse_iterator< _Iterator >, 2711
 - std::slice_array< _Tp >, 2792
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2012
 - std::valarray< _Tp >, 3097
- operator/
 - Complex Numbers, 193
 - std::filesystem::path, 2619
 - Time, 420
- operator/=
 - Complex Numbers, 193
 - std::gslice_array< _Tp >, 2132
 - std::indirect_array< _Tp >, 2178
 - std::mask_array< _Tp >, 2366
 - std::slice_array< _Tp >, 2792

std::valarray< _Tp >, 3098

operator=

- __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 918, 919
- __gnu_debug::__Safe_iterator< _Iterator, _Sequence, _Category >, 1038
- __gnu_debug::__Safe_local_iterator< _Iterator, _Sequence >, 1050
- Complex Numbers, 194
- Numeric Arrays, 233–236
- std::any, 1114
- std::auto_ptr< _Tp >, 1154
- std::back_insert_iterator< _Container >, 1158
- std::basic_regex< _Ch_type, _Rx_traits >, 1537, 1538
- std::basic_string< _CharT, _Traits, _Alloc >, 1640, 1641
- std::deque< _Tp, _Alloc >, 1982, 1983
- std::experimental::fundamentals_v1::any, 1116, 1117
- std::forward_list< _Tp, _Alloc >, 2086
- std::front_insert_iterator< _Container >, 2097
- std::function< _Res(_ArgTypes...)>, 2100–2102
- std::insert_iterator< _Container >, 2184
- std::list< _Tp, _Alloc >, 2299, 2300
- std::locale, 2314
- std::map< _Key, _Tp, _Compare, _Alloc >, 2359, 2360
- std::match_results< _Bi_iter, _Alloc >, 2373
- std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>, 2425, 2426
- std::multimap< _Key, _Tp, _Compare, _Alloc >, 2451, 2452
- std::multiset< _Key, _Compare, _Alloc >, 2476, 2477
- std::once_flag, 2564
- std::ostream_iterator< _Tp, _CharT, _Traits >, 2570
- std::ostreambuf_iterator< _CharT, _Traits >, 2574
- std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2693
- std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2697
- std::set< _Key, _Compare, _Alloc >, 2763
- std::tr2::dynamic_bitset< _WordT, _Alloc >, 2012
- std::unique_ptr< _Tp, _Dp >, 2968, 2969
- std::unique_ptr< _Tp[], _Dp >, 2973
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 3005, 3006
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3034, 3035
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3061, 3062
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3089
- std::vector< _Tp, _Alloc >, 3116, 3117

operator==

- __gnu_cxx, 445, 446
- __gnu_parallel::__IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, 968
- __gnu_pbds::detail::bin_search_tree_const_node_it< _Node, Const_Iterator, Iterator, _Alloc >, 1732
- __gnu_pbds::detail::bin_search_tree_node_it< _Node, Const_Iterator, Iterator, _Alloc >, 1737
- __gnu_pbds::detail::binary_heap_const_iterator< _Value_Type, Entry, Simple, _Alloc >, 1747
- __gnu_pbds::detail::binary_heap_point_const_iterator< _Value_Type, Entry, Simple, _Alloc >, 1750
- __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< _Node, _Alloc >, 2267
- __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< _Node, _Alloc >, 2271
- __gnu_pbds::detail::pat_trie_base::__Node_citer< _Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >, 1003
- __gnu_pbds::detail::pat_trie_base::__Node_iter< _Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >, 1009
- Complex Numbers, 194
- Diagnostics, 133
- Iterators, 178
- Pointer Abstractions, 382, 383
- Regular Expressions, 281–283
- std, 683–688
- std::_Fwd_list_const_iterator< _Tp >, 957
- std::_Fwd_list_iterator< _Tp >, 958
- std::_V2::error_category, 2047
- std::bernoulli_distribution, 1727
- std::binomial_distribution< _IntType >, 1757
- std::bitset< _Nb >, 1771
- std::cauchy_distribution< _RealType >, 1780
- std::chi_squared_distribution< _RealType >, 1801
- std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1995
- std::discrete_distribution< _IntType >, 1999
- std::exponential_distribution< _RealType >, 2056
- std::extreme_value_distribution< _RealType >, 2060
- std::filesystem::path, 2619
- std::fisher_f_distribution< _RealType >, 2069
- std::gamma_distribution< _RealType >, 2112
- std::geometric_distribution< _IntType >, 2114
- std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2175
- std::istream_iterator< _Tp, _CharT, _Traits, _Dist >, 2252
- std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2280
- std::locale, 2314
- std::lognormal_distribution< _RealType >, 2325
- std::mersenne_twister_engine< _UIntType, __w,

- `__n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f` >, [2385](#)
- `std::move_only_function`< `_Res(_ArgTypes...)` `_GLIBCXX_MOF_CV noexcept(_Noex)` >, [2426](#)
- `std::negative_binomial_distribution`< `_IntType` >, [2488](#)
- `std::normal_distribution`< `_RealType` >, [2494](#)
- `std::piecewise_constant_distribution`< `_RealType` >, [2623](#)
- `std::piecewise_linear_distribution`< `_RealType` >, [2626](#)
- `std::poisson_distribution`< `_IntType` >, [2637](#)
- `std::regex_iterator`< `_Bi_iter, _Ch_type, _Rx_traits` >, [2693](#)
- `std::regex_token_iterator`< `_Bi_iter, _Ch_type, _Rx_traits` >, [2698](#)
- `std::shuffle_order_engine`< `_RandomNumberEngine, __k` >, [2788](#)
- `std::slice`, [2790](#)
- `std::student_t_distribution`< `_RealType` >, [2848](#)
- `std::sub_match`< `_Biter` >, [2856](#)
- `std::subtract_with_carry_engine`< `_UIntType, __w, __s, __r` >, [2861](#)
- `std::uniform_int_distribution`< `_IntType` >, [2960](#)
- `std::uniform_real_distribution`< `_RealType` >, [2963](#)
- `std::weibull_distribution`< `_RealType` >, [3145](#)
- Time, [421](#)
- Utilities, [348](#)
- `operator""d`
 - Time, [417](#)
- `operator""h`
 - Time, [417](#)
- `operator""min`
 - Time, [417](#)
- `operator""ms`
 - Time, [418](#)
- `operator""ns`
 - Time, [418](#)
- `operator""s`
 - Time, [418](#)
- `operator""us`
 - Time, [418](#)
- `operator""y`
 - Time, [418](#)
- `operator%`
 - Time, [419](#)
- `operator%=>`
 - `std::gslice_array`< `_Tp` >, [2131](#)
 - `std::indirect_array`< `_Tp` >, [2177](#)
 - `std::mask_array`< `_Tp` >, [2365](#)
 - `std::slice_array`< `_Tp` >, [2792](#)
 - `std::valarray`< `_Tp` >, [3096](#)
- `operator&`
 - Dynamic Bitset., [142](#)
- `std`, [663](#)
- `std::regex_constants`, [777](#), [778](#)
- `operator&=`
 - `std::bitset`< `_Nb` >, [1770](#)
 - `std::gslice_array`< `_Tp` >, [2131](#)
 - `std::indirect_array`< `_Tp` >, [2177](#)
 - `std::mask_array`< `_Tp` >, [2366](#)
 - `std::regex_constants`, [778](#)
 - `std::slice_array`< `_Tp` >, [2792](#)
 - `std::tr2::dynamic_bitset`< `_WordT, _Alloc` >, [2011](#)
 - `std::valarray`< `_Tp` >, [3096](#)
- `operator[]`
 - `__gnu_cxx::__versa_string`< `_CharT, _Traits, _Alloc, _Base` >, [919](#), [920](#)
- Numeric Arrays, [236–239](#)
- `std::basic_string`< `_CharT, _Traits, _Alloc` >, [1641](#), [1642](#)
- `std::bitset`< `_Nb` >, [1772](#)
- `std::deque`< `_Tp, _Alloc` >, [1983](#), [1984](#)
- `std::map`< `_Key, _Tp, _Compare, _Alloc` >, [2360](#)
- `std::match_results`< `_Bi_iter, _Alloc` >, [2373](#)
- `std::reverse_iterator`< `_Iterator` >, [2712](#)
- `std::tr2::dynamic_bitset`< `_WordT, _Alloc` >, [2013](#)
- `std::unique_ptr`< `_Tp[]`, `_Dp` >, [2974](#)
- `std::unordered_map`< `_Key, _Tp, _Hash, _Pred, _Alloc` >, [3006](#)
- `std::vector`< `_Tp, _Alloc` >, [3117](#), [3118](#)
- `operator|`
 - Dynamic Bitset., [144](#)
 - `std`, [699](#)
 - `std::regex_constants`, [779](#)
- `operator|=`
 - `std::bitset`< `_Nb` >, [1773](#)
 - `std::gslice_array`< `_Tp` >, [2132](#)
 - `std::indirect_array`< `_Tp` >, [2178](#)
 - `std::mask_array`< `_Tp` >, [2367](#)
 - `std::regex_constants`, [779](#)
 - `std::slice_array`< `_Tp` >, [2793](#)
 - `std::tr2::dynamic_bitset`< `_WordT, _Alloc` >, [2014](#)
 - `std::valarray`< `_Tp` >, [3099](#)
- `opt_random.h`, [3593](#)
- `optimize`
 - `std::regex_constants`, [783](#)
- optional, [3491](#), [3492](#)
- Optional values, [315](#)
 - `in_place`, [316](#)
 - `nullopt`, [316](#)
- `order_of_key`
 - `__gnu_pbds::tree_order_statistics_node_update`< `Node_Cltr, Node_Itr, Cmp_Fn, _Alloc` >, [2917](#)
 - `__gnu_pbds::trie_order_statistics_node_update`< `Node_Cltr, Node_Itr, _ATraits, _Alloc` >, [2930](#)
- `order_of_prefix`

- `__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >`, 2930
- `order_preserving`
 - `__gnu_pbds::container_traits< Cntr >`, 1878
- `order_statistics_imp.hpp`, 3441
- `os_defines.h`, 3593
- `ostream`, 3495
 - I/O, 171
- `ostream.tcc`, 3230
- `ostream_insert.h`, 3230
- `ostream_iterator`
 - `std::ostream_iterator< _Tp, _CharT, _Traits >`, 2570
- `ostream_type`
 - `std::ostream_iterator< _Tp, _CharT, _Traits >`, 2569
 - `std::ostreambuf_iterator< _CharT, _Traits >`, 2573
- `ostreambuf_iterator`
 - `std::ostreambuf_iterator< _CharT, _Traits >`, 2574
- `ostreamstringstream`
 - I/O, 171
- `out`
 - `std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >`, 820
 - `std::basic_fstream< _CharT, _Traits >`, 1239
 - `std::basic_ifstream< _CharT, _Traits >`, 1282
 - `std::basic_ios< _CharT, _Traits >`, 1305
 - `std::basic_iostream< _CharT, _Traits >`, 1351
 - `std::basic_istream< _CharT, _Traits >`, 1388
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1430
 - `std::basic_ofstream< _CharT, _Traits >`, 1464
 - `std::basic_ostream< _CharT, _Traits >`, 1496
 - `std::basic_ostringstream< _CharT, _Traits, _Alloc >`, 1528
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1724
 - `std::codecvt< _InternT, _ExternT, _StateT >`, 1806
 - `std::codecvt< _InternT, _ExternT, encoding_state >`, 1811
 - `std::codecvt< char, char, mbstate_t >`, 1816
 - `std::codecvt< char16_t, char, mbstate_t >`, 1820
 - `std::codecvt< char32_t, char, mbstate_t >`, 1825
 - `std::codecvt< wchar_t, char, mbstate_t >`, 1830
 - `std::codecvt_byname< _InternT, _ExternT, _StateT >`, 1836
 - `std::ios_base`, 2201
- `ov_tree_map.hpp`, 3433
- `overflow`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2028
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2812
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2829
 - `std::basic_filebuf< _CharT, _Traits >`, 1178
 - `std::basic_streambuf< _CharT, _Traits >`, 1545
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1666
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3131
- `owner_before`
 - `std::shared_ptr< _Tp >`, 2783
- `p`
 - `std::bernoulli_distribution`, 1727
 - `std::binomial_distribution< _IntType >`, 1756
 - `std::geometric_distribution< _IntType >`, 2114
 - `std::negative_binomial_distribution< _IntType >`, 2487
- `pair`
 - `std::pair< _T1, _T2 >`, 2592
 - Utilities, 340
- `pairing_heap.hpp`, 3433
- `par_loop.h`, 3522
- `parallel.h`, 3523
- `parallel_balanced`
 - `__gnu_parallel`, 468
- `parallel_multiway_merge`
 - `__gnu_parallel`, 506
- `parallel_omp_loop`
 - `__gnu_parallel`, 468
- `parallel_omp_loop_static`
 - `__gnu_parallel`, 468
- `parallel_sort_mwms`
 - `__gnu_parallel`, 507
- `parallel_sort_mwms_pu`
 - `__gnu_parallel`, 507
- `parallel_tag`
 - `__gnu_parallel::parallel_tag`, 2598
- `parallel_taskqueue`
 - `__gnu_parallel`, 468
- `parallel_unbalanced`
 - `__gnu_parallel`, 468
- Parallelism TS, 322
- `param`
 - `std::bernoulli_distribution`, 1727
 - `std::binomial_distribution< _IntType >`, 1756
 - `std::cauchy_distribution< _RealType >`, 1780
 - `std::chi_squared_distribution< _RealType >`, 1800
 - `std::discrete_distribution< _IntType >`, 1998
 - `std::exponential_distribution< _RealType >`, 2055, 2056
 - `std::extreme_value_distribution< _RealType >`, 2059
 - `std::fisher_f_distribution< _RealType >`, 2068
 - `std::gamma_distribution< _RealType >`, 2111
 - `std::geometric_distribution< _IntType >`, 2114
 - `std::lognormal_distribution< _RealType >`, 2324
 - `std::negative_binomial_distribution< _IntType >`, 2487
 - `std::normal_distribution< _RealType >`, 2493, 2494
 - `std::piecewise_constant_distribution< _RealType >`, 2622

- std::piecewise_linear_distribution< _RealType >, 2625
- std::poisson_distribution< _IntType >, 2636
- std::student_t_distribution< _RealType >, 2848
- std::uniform_int_distribution< _IntType >, 2959, 2960
- std::uniform_real_distribution< _RealType >, 2962
- std::weibull_distribution< _RealType >, 3144
- parse_numbers.h, 3231
- partial_sort
 - Sorting, 80, 81
- partial_sort_copy
 - Sorting, 82
- partial_sort_minimal_n
 - __gnu_parallel::Settings, 1077
- partial_sum
 - Generalized Numeric operations, 13
- partial_sum.h, 3523
- partial_sum_dilation
 - __gnu_parallel::Settings, 1077
- partial_sum_minimal_n
 - __gnu_parallel::Settings, 1077
- partition
 - Mutating, 29
- partition.h, 3523
 - _GLIBCXX_VOLATILE, 3524
- partition_chunk_share
 - __gnu_parallel::Settings, 1077
- partition_chunk_size
 - __gnu_parallel::Settings, 1077
- partition_copy
 - Mutating, 29
- partition_minimal_n
 - __gnu_parallel::Settings, 1077
- partition_point
 - Mutating, 30
- pat_trie_.hpp, 3434
- pat_trie_base.hpp, 3434
- pbackfail
 - __gnu_cxx::enc_filebuf< _CharT >, 2028
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2813
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2830
 - std::basic_filebuf< _CharT, _Traits >, 1179
 - std::basic_streambuf< _CharT, _Traits >, 1546
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1666
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3131
- pbase
 - __gnu_cxx::enc_filebuf< _CharT >, 2028
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2813
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2830
 - std::basic_filebuf< _CharT, _Traits >, 1179
 - std::basic_streambuf< _CharT, _Traits >, 1546
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1667
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3132
- pbump
 - __gnu_cxx::enc_filebuf< _CharT >, 2029
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2814
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2830
 - std::basic_filebuf< _CharT, _Traits >, 1179
 - std::basic_streambuf< _CharT, _Traits >, 1546
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1667
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3132
- peek
 - std::basic_fstream< _CharT, _Traits >, 1224
 - std::basic_ifstream< _CharT, _Traits >, 1269
 - std::basic_iostream< _CharT, _Traits >, 1336
 - std::basic_istream< _CharT, _Traits >, 1376
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1415
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1709
- perm_options
 - File System, 168
- perms
 - File System, 168
 - Filesystem TS, 308
- phi_v
 - std::numbers, 773
- pi_v
 - std::numbers, 773
- piecewise_construct
 - Utilities, 350
- pod_char_traits.h, 3447
- point_const_iterator.hpp, 3417
- point_iterator.hpp, 3443
- point_iterators.hpp, 3404
- pointer
 - __gnu_pbds::detail::binary_heap_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1745
 - __gnu_pbds::detail::binary_heap_point_const_iterator< Value_Type, Entry, Simple, _Alloc >, 1748
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc >, 2266
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc >, 2269
 - std::allocator_traits< _Alloc >, 1094
 - std::allocator_traits< allocator< _Tp > >, 1099
 - std::allocator_traits< allocator< void > >, 1104
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, 1107
 - std::back_insert_iterator< _Container >, 1157
 - std::front_insert_iterator< _Container >, 2096
 - std::insert_iterator< _Container >, 2183
 - std::istream_iterator< _Tp, _CharT, _Traits, _Dist >, 2251

- std::istreambuf_iterator< _CharT, _Traits >, [2254](#)
- std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >, [2260](#)
- std::ostream_iterator< _Tp, _CharT, _Traits >, [2569](#)
- std::ostreambuf_iterator< _CharT, _Traits >, [2573](#)
- std::pointer_traits< _Ptr >, [2632](#)
- std::pointer_traits< _Tp * >, [2633](#)
- std::raw_storage_iterator< _OutputIterator, _Tp >, [2671](#)
- std::set< _Key, _Compare, _Alloc >, [2747](#)
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2984](#)
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3017](#)
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3044](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3071](#)
- Pointer Abstractions, [369](#)
 - allocate_shared, [373](#)
 - atomic_compare_exchange_strong_explicit, [374](#)
 - atomic_exchange_explicit, [374](#)
 - atomic_is_lock_free, [375](#)
 - atomic_load_explicit, [375](#)
 - atomic_store_explicit, [375](#)
 - const_pointer_cast, [376](#)
 - dynamic_pointer_cast, [376](#)
 - get_deleter, [376](#)
 - make_shared, [377](#)
 - make_unique, [377](#), [378](#)
 - make_unique_for_overwrite, [378](#), [379](#)
 - operator!=, [379](#), [380](#)
 - operator<, [380](#), [381](#)
 - operator<<, [381](#)
 - operator<=, [382](#)
 - operator>, [383](#), [384](#)
 - operator>=, [384](#), [385](#)
 - operator==, [382](#), [383](#)
 - reinterpret_pointer_cast, [385](#)
 - static_pointer_cast, [385](#), [386](#)
 - swap, [386](#)
 - to_address, [386](#), [387](#)
- Pointer Safety and Garbage Collection, [387](#)
 - declare_no_pointers, [388](#)
 - declare_reachable, [388](#)
 - get_pointer_safety, [388](#)
 - pointer_safety, [388](#)
 - undecare_no_pointers, [388](#)
 - undecare_reachable, [388](#)
- pointer.h, [3448](#)
- pointer_safety
 - Pointer Safety and Garbage Collection, [388](#)
- pointer_to
 - std::pointer_traits< _Ptr >, [2632](#)
 - std::pointer_traits< _Tp * >, [2634](#)
- Poisson Distributions, [249](#)
- operator!=, [250](#), [251](#)
- operator<<, [251](#), [252](#)
- operator>>, [252](#), [253](#)
- polar
 - Complex Numbers, [194](#)
- Policy-Based Data Structures, [145](#)
- policy_access_fn_imps.hpp, [3418](#)
- pool_allocator.h, [3449](#)
- pop
 - std::priority_queue< _Tp, _Sequence, _Compare >, [2643](#)
 - std::queue< _Tp, _Sequence >, [2652](#)
 - std::stack< _Tp, _Sequence >, [2803](#)
- pop_back
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [920](#)
 - __gnu_parallel::__RestrictedBoundedConcurrentQueue< _Tp >, [1025](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1642](#)
 - std::deque< _Tp, _Alloc >, [1984](#)
 - std::list< _Tp, _Alloc >, [2300](#)
 - std::vector< _Tp, _Alloc >, [3118](#)
- pop_front
 - __gnu_parallel::__RestrictedBoundedConcurrentQueue< _Tp >, [1025](#)
 - std::deque< _Tp, _Alloc >, [1984](#)
 - std::forward_list< _Tp, _Alloc >, [2087](#)
 - std::list< _Tp, _Alloc >, [2300](#)
- pop_heap
 - Heap, [95](#)
- popcount
 - Bit manipulation, [185](#)
- pos_format
 - std::moneypunct< _CharT, _Intl >, [2413](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2421](#)
- pos_type
 - std::basic_ios< _CharT, _Traits >, [1290](#)
 - std::basic_streambuf< _CharT, _Traits >, [1542](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3128](#)
- position
 - std::match_results< _Bi_iter, _Alloc >, [2374](#)
- positive_sign
 - std::moneypunct< _CharT, _Intl >, [2413](#)
 - std::moneypunct_byname< _CharT, _Intl >, [2421](#)
- postypes.h, [3231](#)
- pow
 - Complex Numbers, [194](#), [195](#)
- power
 - SGL, [161](#), [162](#)
- pptr
 - __gnu_cxx::enc_filebuf< _CharT >, [2029](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2814](#)

- `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2831
- `std::basic_filebuf< _CharT, _Traits >`, 1180
- `std::basic_streambuf< _CharT, _Traits >`, 1547
- `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1667
- `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3132
- precision
 - `std::basic_fstream< _CharT, _Traits >`, 1224
 - `std::basic_ifstream< _CharT, _Traits >`, 1269
 - `std::basic_ios< _CharT, _Traits >`, 1296
 - `std::basic_iostream< _CharT, _Traits >`, 1336
 - `std::basic_istream< _CharT, _Traits >`, 1376
 - `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1415
 - `std::basic_ofstream< _CharT, _Traits >`, 1452
 - `std::basic_ostream< _CharT, _Traits >`, 1485
 - `std::basic_ostreamstream< _CharT, _Traits, _Alloc >`, 1516
 - `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1709
 - `std::ios_base`, 2193
- `predefined_ops.h`, 3232
- prefix
 - `std::match_results< _Bi_iter, _Alloc >`, 2374
- `prefix_range`
 - `__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >`, 2934, 2935
- `prefix_search_node_update_imp.hpp`, 3441
- `prev_permutation`
 - Sorting, 83
- `priority_queue`
 - `__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >`, 2640
 - `std::priority_queue< _Tp, _Sequence, _Compare >`, 2642
- `priority_queue.hpp`, 3445
- `priority_queue_base_dispatch.hpp`, 3435
- probabilities
 - `std::discrete_distribution< _IntType >`, 1998
- `probe_fn_base.hpp`, 3428
- `propagate_const`, 3396
- `propagate_on_container_copy_assignment`
 - `__gnu_cxx::__alloc_traits< _Alloc, typename >`, 806
 - `std::allocator_traits< _Alloc >`, 1095
 - `std::allocator_traits< allocator< _Tp > >`, 1099
 - `std::allocator_traits< allocator< void > >`, 1104
 - `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, 1107
- `propagate_on_container_move_assignment`
 - `__gnu_cxx::__alloc_traits< _Alloc, typename >`, 806
 - `std::allocator_traits< _Alloc >`, 1095
 - `std::allocator_traits< allocator< _Tp > >`, 1099
 - `std::allocator_traits< allocator< void > >`, 1104
- `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, 1107
- `propagate_on_container_swap`
 - `__gnu_cxx::__alloc_traits< _Alloc, typename >`, 806
 - `std::allocator_traits< _Alloc >`, 1095
 - `std::allocator_traits< allocator< _Tp > >`, 1099
 - `std::allocator_traits< allocator< void > >`, 1104
 - `std::allocator_traits< pmr::polymorphic_allocator< _Tp > >`, 1107
- `ptr_fun`
 - Adaptors for pointers to functions, 353, 354
- `ptr_traits.h`, 3233
- `pubimbue`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2029
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2814
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2831
 - `std::basic_filebuf< _CharT, _Traits >`, 1180
 - `std::basic_streambuf< _CharT, _Traits >`, 1547
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1667
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3132
- `pubseekoff`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2030
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2815
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2831
 - `std::basic_filebuf< _CharT, _Traits >`, 1180
 - `std::basic_streambuf< _CharT, _Traits >`, 1547
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1668
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3133
- `pubseekpos`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2030
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2815
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2832
 - `std::basic_filebuf< _CharT, _Traits >`, 1180
 - `std::basic_streambuf< _CharT, _Traits >`, 1547
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1668
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3133
- `pubsetbuf`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2030
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2815
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2832
 - `std::basic_filebuf< _CharT, _Traits >`, 1181
 - `std::basic_streambuf< _CharT, _Traits >`, 1548
 - `std::basic_stringbuf< _CharT, _Traits, _Alloc >`, 1668
 - `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, 3133
- `pubsync`
 - `__gnu_cxx::enc_filebuf< _CharT >`, 2030
 - `__gnu_cxx::stdio_filebuf< _CharT, _Traits >`, 2815
 - `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >`, 2832
 - `std::basic_filebuf< _CharT, _Traits >`, 1181

- std::basic_streambuf< _CharT, _Traits >, 1548
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1668
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3133
- push
 - std::priority_queue< _Tp, _Sequence, _Compare >, 2643
 - std::queue< _Tp, _Sequence >, 2652
 - std::stack< _Tp, _Sequence >, 2803
- push_back
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 920
 - std::basic_string< _CharT, _Traits, _Alloc >, 1642
 - std::deque< _Tp, _Alloc >, 1984
 - std::list< _Tp, _Alloc >, 2300
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2014
 - std::vector< _Tp, _Alloc >, 3118
- push_front
 - __gnu_parallel::__RestrictedBoundedConcurrentQueue< _Tp >, 1025
 - std::deque< _Tp, _Alloc >, 1985
 - std::forward_list< _Tp, _Alloc >, 2087
 - std::list< _Tp, _Alloc >, 2301
- push_heap
 - Heap, 95, 96
- put
 - std::basic_fstream< _CharT, _Traits >, 1225
 - std::basic_istream< _CharT, _Traits >, 1337
 - std::basic_ofstream< _CharT, _Traits >, 1454
 - std::basic_ostream< _CharT, _Traits >, 1485
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1516
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1710
 - std::money_put< _CharT, _OutIter >, 2404
 - std::num_put< _CharT, _OutIter >, 2522–2527
 - std::time_put< _CharT, _OutIter >, 2906, 2907
 - std::time_put_byname< _CharT, _OutIter >, 2909, 2910
- put_money
 - std, 699
- put_time
 - std, 700
- putback
 - std::basic_fstream< _CharT, _Traits >, 1225
 - std::basic_ifstream< _CharT, _Traits >, 1270
 - std::basic_istream< _CharT, _Traits >, 1337
 - std::basic_istream< _CharT, _Traits >, 1376
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1415
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1710
- pword
 - std::basic_fstream< _CharT, _Traits >, 1226
 - std::basic_ifstream< _CharT, _Traits >, 1270
- std::basic_ios< _CharT, _Traits >, 1297
- std::basic_iostream< _CharT, _Traits >, 1338
- std::basic_istream< _CharT, _Traits >, 1377
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1416
- std::basic_ofstream< _CharT, _Traits >, 1454
- std::basic_ostream< _CharT, _Traits >, 1486
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1517
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1711
- std::ios_base, 2194
- qsb_steals
 - __gnu_parallel::__Settings, 1077
- quadratic_probe_fn_imp.hpp, 3429
- queue, 3530
 - std::queue< _Tp, _Sequence >, 2652
- queue.h, 3524
 - _GLIBCXX_VOLATILE, 3524
- quicksort.h, 3524
- quiet_NaN
 - std::numeric_limits< _Tp >, 2530
- quoted
 - std, 700
- quoted_string.h, 3233
- r_erase_fn_imps.hpp, 3405
- radix
 - std::__numeric_limits_base, 865
 - std::numeric_limits< _Tp >, 2533
- random, 3531
- Random Number Distributions, 243
- Random Number Generation, 242
 - generate_canonical, 242
- Random Number Generators, 256
 - minstd_rand, 258
 - minstd_rand0, 258
 - mt19937, 258
 - mt19937_64, 258
 - operator!=, 258–260
 - operator<=, 260
- Random Number Utilities, 261
- random.h, 3234
- random.tcc, 3238, 3241
- random_number.h, 3525
- random_sample
 - SGI, 162
- random_sample_n
 - SGI, 162
- random_shuffle
 - Mutating, 30, 31
- random_shuffle.h, 3525
- random_shuffle_minimal_n
 - __gnu_parallel::__Settings, 1077

- range_access.h, [3244](#)
- ranged_hash_fn.hpp, [3429](#)
- ranged_probe_fn.hpp, [3429](#)
- Ranges, [267](#)
- ranges, [3531](#)
- ranges_algo.h, [3245](#)
- ranges_algobase.h, [3248](#)
- ranges_base.h, [3249](#)
 - enable_view, [3251](#)
- ranges_cmp.h, [3252](#)
- ranges_uninitialized.h, [3252](#)
- ranges_util.h, [3252](#)
- ratio, [3534–3536](#)
- ratio_add
 - Rational Arithmetic, [407](#)
- ratio_divide
 - Rational Arithmetic, [407](#)
- ratio_multiply
 - Rational Arithmetic, [407](#)
- ratio_subtract
 - Rational Arithmetic, [408](#)
- Rational Arithmetic, [406](#)
 - ratio_add, [407](#)
 - ratio_divide, [407](#)
 - ratio_multiply, [407](#)
 - ratio_subtract, [408](#)
- rb_tree, [3450](#)
- rb_tree.hpp, [3436](#)
- rbegin
 - gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>, [921](#)
 - std, [700](#), [701](#)
 - std::basic_string<_CharT, _Traits, _Alloc>, [1643](#)
 - std::deque<_Tp, _Alloc>, [1985](#)
 - std::list<_Tp, _Alloc>, [2301](#)
 - std::map<_Key, _Tp, _Compare, _Alloc>, [2360](#)
 - std::multimap<_Key, _Tp, _Compare, _Alloc>, [2452](#)
 - std::multiset<_Key, _Compare, _Alloc>, [2477](#)
 - std::set<_Key, _Compare, _Alloc>, [2763](#)
 - std::vector<_Tp, _Alloc>, [3118](#), [3119](#)
- rc.hpp, [3436](#)
- rc_binomial_heap.hpp, [3436](#)
- rc_string_base.h, [3450](#)
- rdbuf
 - std::basic_fstream<_CharT, _Traits>, [1226](#)
 - std::basic_ifstream<_CharT, _Traits>, [1271](#)
 - std::basic_ios<_CharT, _Traits>, [1297](#)
 - std::basic_iostream<_CharT, _Traits>, [1338](#)
 - std::basic_istream<_CharT, _Traits>, [1377](#), [1378](#)
 - std::basic_istreamstream<_CharT, _Traits, _Alloc>, [1416](#)
 - std::basic_ofstream<_CharT, _Traits>, [1454](#), [1455](#)
 - std::basic_ostream<_CharT, _Traits>, [1486](#), [1487](#)
 - std::basic_ostreamstream<_CharT, _Traits, _Alloc>, [1517](#)
 - std::basic_stringstream<_CharT, _Traits, _Alloc>, [1711](#)
- rdstate
 - std::basic_fstream<_CharT, _Traits>, [1227](#)
 - std::basic_ifstream<_CharT, _Traits>, [1271](#)
 - std::basic_ios<_CharT, _Traits>, [1298](#)
 - std::basic_iostream<_CharT, _Traits>, [1339](#)
 - std::basic_istream<_CharT, _Traits>, [1378](#)
 - std::basic_istreamstream<_CharT, _Traits, _Alloc>, [1418](#)
 - std::basic_ofstream<_CharT, _Traits>, [1455](#)
 - std::basic_ostream<_CharT, _Traits>, [1487](#)
 - std::basic_ostreamstream<_CharT, _Traits, _Alloc>, [1518](#)
 - std::basic_stringstream<_CharT, _Traits, _Alloc>, [1712](#)
- read
 - std::basic_fstream<_CharT, _Traits>, [1227](#)
 - std::basic_ifstream<_CharT, _Traits>, [1271](#)
 - std::basic_iostream<_CharT, _Traits>, [1339](#)
 - std::basic_istream<_CharT, _Traits>, [1378](#)
 - std::basic_istreamstream<_CharT, _Traits, _Alloc>, [1418](#)
 - std::basic_stringstream<_CharT, _Traits, _Alloc>, [1712](#)
- readsome
 - std::basic_fstream<_CharT, _Traits>, [1227](#)
 - std::basic_ifstream<_CharT, _Traits>, [1272](#)
 - std::basic_iostream<_CharT, _Traits>, [1340](#)
 - std::basic_istream<_CharT, _Traits>, [1379](#)
 - std::basic_istreamstream<_CharT, _Traits, _Alloc>, [1419](#)
 - std::basic_stringstream<_CharT, _Traits, _Alloc>, [1712](#)
- ready
 - std::match_results<_Bi_iter, _Alloc>, [2374](#)
- rebind
 - std::pointer_traits<_Ptr>, [2632](#)
 - std::pointer_traits<_Tp*>, [2633](#)
- reduce
 - Generalized Numeric operations, [15](#), [16](#)
- ref
 - std::reference_wrapper<_Tp>, [2689](#), [2690](#)
- reference
 - gnu_pbds::detail::bin_search_tree_const_node_it<_Node, Const_Iterator, Iterator, _Alloc>, [1731](#)
 - gnu_pbds::detail::bin_search_tree_node_it<_Node, Const_Iterator, Iterator, _Alloc>, [1736](#)
 - gnu_pbds::detail::binary_heap_const_iterator<_Value_Type, Entry, Simple, _Alloc>, [1745](#)
 - gnu_pbds::detail::binary_heap_point_const_iterator<_Value_Type, Entry, Simple, _Alloc>, [1749](#)

- `__gnu_pbds::detail::left_child_next_sibling_heap_constructor_traits`
- `Node, _Alloc >`, 2266
- `__gnu_pbds::detail::left_child_next_sibling_heap_node_registration_traits`
- `Node, _Alloc >`, 2270
- `std::back_insert_iterator< _Container >`, 1157
- `std::front_insert_iterator< _Container >`, 2096
- `std::insert_iterator< _Container >`, 2183
- `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >`, 2251
- `std::istreambuf_iterator< _CharT, _Traits >`, 2254
- `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`, 2260
- `std::ostream_iterator< _Tp, _CharT, _Traits >`, 2569
- `std::ostreambuf_iterator< _CharT, _Traits >`, 2573
- `std::raw_storage_iterator< _OutputIterator, _Tp >`, 2671
- `std::set< _Key, _Compare, _Alloc >`, 2747
- `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >`, 2985
- `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3017
- `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`, 3044
- `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`, 3071
- `refwrap.h`, 3253
- `regex`, 3536
 - Regular Expressions, 272
- `regex.h`, 3254
- `regex.tcc`, 3256
- `regex_automaton.h`, 3256
- `regex_automaton.tcc`, 3257
- `regex_compiler.h`, 3257
- `regex_compiler.tcc`, 3258
- `regex_constants.h`, 3258
- `regex_error`
 - `std::regex_error`, 2691
- `regex_error.h`, 3259
- `regex_executor.h`, 3260
- `regex_executor.tcc`, 3260
- `regex_iterator`
 - `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >`, 2692
- `regex_match`
 - Regular Expressions, 288–292
- `regex_replace`
 - Regular Expressions, 292–295
- `regex_scanner.h`, 3261
- `regex_scanner.tcc`, 3261
- `regex_search`
 - Regular Expressions, 296–299
- `regex_token_iterator`
 - `std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >`, 2695, 2696
- `std::regex_traits< _Ch_type >`, 2699
- `std::basic_fstream< _CharT, _Traits >`, 1228
- `std::basic_ifstream< _CharT, _Traits >`, 1273
- `std::basic_ios< _CharT, _Traits >`, 1298
- `std::basic_iostream< _CharT, _Traits >`, 1340
- `std::basic_istream< _CharT, _Traits >`, 1379
- `std::basic_istreamstream< _CharT, _Traits, _Alloc >`, 1419
- `std::basic_ofstream< _CharT, _Traits >`, 1455
- `std::basic_ostream< _CharT, _Traits >`, 1487
- `std::basic_ostringstream< _CharT, _Traits, _Alloc >`, 1518
- `std::basic_stringstream< _CharT, _Traits, _Alloc >`, 1713
- `std::ios_base`, 2194
- Regular Expressions, 267
 - `__regex_replace`, 273
 - `cregex_token_iterator`, 272
 - `csub_match`, 272
 - `operator!=`, 273–276
 - `operator<`, 276–278
 - `operator<<`, 278
 - `operator<=`, 279–281
 - `operator>`, 284–286
 - `operator>=`, 286–288
 - `operator==`, 281–283
 - `regex`, 272
 - `regex_match`, 288–292
 - `regex_replace`, 292–295
 - `regex_search`, 296–299
 - `sregex_token_iterator`, 272
 - `ssub_match`, 272
 - `swap`, 300
 - `wcregex_token_iterator`, 272
 - `wcsub_match`, 272
 - `wregex`, 272
 - `wsregex_token_iterator`, 272
 - `wssub_match`, 273
- `rehash`
 - `std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3006
 - `std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >`, 3035
 - `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`, 3062
 - `std::unordered_set< _Value, _Hash, _Pred, _Alloc >`, 3090
- `reinterpret_pointer_cast`
 - Pointer Abstractions, 385
- `release`
 - `std::auto_ptr< _Tp >`, 1155
 - `std::unique_ptr< _Tp, _Dp >`, 2969

- std::unique_ptr< _Tp[], _Dp >, 2974
- remove
 - Mutating, 31
 - std::forward_list< _Tp, _Alloc >, 2087
 - std::list< _Tp, _Alloc >, 2301
- remove_all_extents_t
 - Metaprogramming, 398
- remove_const_t
 - Metaprogramming, 398
- remove_copy
 - Mutating, 32
- remove_copy_if
 - Mutating, 32
- remove_cv_t
 - Metaprogramming, 399
- remove_cvref_t
 - Metaprogramming, 399
- remove_extent_t
 - Metaprogramming, 399
- remove_if
 - Mutating, 33
 - std::forward_list< _Tp, _Alloc >, 2087
 - std::list< _Tp, _Alloc >, 2302
- remove_pointer_t
 - Metaprogramming, 399
- remove_reference_t
 - Metaprogramming, 399
- remove_volatile_t
 - Metaprogramming, 399
- rend
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 921
 - std, 701, 702
 - std::basic_string< _CharT, _Traits, _Alloc >, 1643
 - std::deque< _Tp, _Alloc >, 1985
 - std::list< _Tp, _Alloc >, 2302
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2361
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2452
 - std::multiset< _Key, _Compare, _Alloc >, 2477
 - std::set< _Key, _Compare, _Alloc >, 2764
 - std::vector< _Tp, _Alloc >, 3119
- replace
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 921–927
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1582–1587
 - Mutating, 33
 - std::basic_string< _CharT, _Traits, _Alloc >, 1643–1651
- replace_copy
 - std, 702
- replace_copy_if
 - Mutating, 34
- replace_if
 - Mutating, 35
- replace_minimal_n
 - __gnu_parallel::Settings, 1077
- requested_size
 - __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, 2869
 - std::Temporary_buffer< _ForwardIterator, _Tp >, 1083
- reserve
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 928
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1588
 - std::basic_string< _CharT, _Traits, _Alloc >, 1651
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 3007
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3035
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3062
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3090
 - std::vector< _Tp, _Alloc >, 3119
- reset
 - std::any, 1114
 - std::auto_ptr< _Tp >, 1155
 - std::bernoulli_distribution, 1727
 - std::binomial_distribution< _IntType >, 1756
 - std::bitset< _Nb >, 1773
 - std::cauchy_distribution< _RealType >, 1780
 - std::chi_squared_distribution< _RealType >, 1801
 - std::discrete_distribution< _IntType >, 1998
 - std::exponential_distribution< _RealType >, 2056
 - std::extreme_value_distribution< _RealType >, 2059
 - std::fisher_f_distribution< _RealType >, 2068
 - std::gamma_distribution< _RealType >, 2111
 - std::geometric_distribution< _IntType >, 2114
 - std::lognormal_distribution< _RealType >, 2324
 - std::negative_binomial_distribution< _IntType >, 2487
 - std::normal_distribution< _RealType >, 2494
 - std::piecewise_constant_distribution< _RealType >, 2622
 - std::piecewise_linear_distribution< _RealType >, 2626
 - std::poisson_distribution< _IntType >, 2636
 - std::student_t_distribution< _RealType >, 2848
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2014
 - std::uniform_int_distribution< _IntType >, 2960
 - std::uniform_real_distribution< _RealType >, 2962
 - std::unique_ptr< _Tp, _Dp >, 2969
 - std::unique_ptr< _Tp[], _Dp >, 2974
 - std::weibull_distribution< _RealType >, 3144

- resetiosflags
 - std, [702](#)
- resize
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [929](#)
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2168](#)
 - Numeric Arrays, [240](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1652](#)
 - std::deque< _Tp, _Alloc >, [1985](#), [1986](#)
 - std::forward_list< _Tp, _Alloc >, [2088](#)
 - std::list< _Tp, _Alloc >, [2302](#), [2303](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2016](#)
 - std::vector< _Tp, _Alloc >, [3119](#), [3120](#)
- resize_fn_imps.hpp, [3425](#)
- resize_no_store_hash_fn_imps.hpp, [3425](#)
- resize_policy.hpp, [3418](#)
- resize_store_hash_fn_imps.hpp, [3425](#)
- result_of_t
 - Metaprogramming, [399](#)
- result_type
 - __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >, [1740](#)
 - __gnu_cxx::project1st< _Arg1, _Arg2 >, [2645](#)
 - __gnu_cxx::project2nd< _Arg1, _Arg2 >, [2646](#)
 - __gnu_cxx::select1st< _Pair >, [2736](#)
 - __gnu_cxx::select2nd< _Pair >, [2736](#)
 - __gnu_cxx::subtractive_rng, [2863](#)
 - __gnu_cxx::unary_compose< _Operation1, _Operation2 >, [2953](#)
 - __gnu_parallel::_EqualFromLess< _T1, _T2, _Compare >, [950](#)
 - __gnu_parallel::_EqualTo< _T1, _T2 >, [951](#)
 - __gnu_parallel::_Less< _T1, _T2 >, [973](#)
 - __gnu_parallel::_Lexicographic< _T1, _T2, _Compare >, [974](#)
 - __gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare >, [975](#)
 - __gnu_parallel::_Multiplies< _Tp1, _Tp2, _Result >, [999](#)
 - __gnu_parallel::_Plus< _Tp1, _Tp2, _Result >, [1014](#)
 - __gnu_parallel::_binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >, [816](#)
 - __gnu_parallel::_binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >, [817](#)
 - __gnu_parallel::_unary_negate< _Predicate, argument_type >, [881](#)
 - std::bernoulli_distribution, [1726](#)
 - std::binary_function< _Arg1, _Arg2, _Result >, [1741](#)
 - std::binary_negate< _Predicate >, [1751](#)
 - std::binder1st< _Operation >, [1752](#)
 - std::binder2nd< _Operation >, [1754](#)
 - std::binomial_distribution< _IntType >, [1755](#)
 - std::cauchy_distribution< _RealType >, [1779](#)
 - std::chi_squared_distribution< _RealType >, [1800](#)
 - std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >, [1861](#)
 - std::const_mem_fun1_t< _Ret, _Tp, _Arg >, [1862](#)
 - std::const_mem_fun_ref_t< _Ret, _Tp >, [1863](#)
 - std::const_mem_fun_t< _Ret, _Tp >, [1864](#)
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, [1992](#)
 - std::discrete_distribution< _IntType >, [1997](#)
 - std::divides< _Tp >, [2000](#)
 - std::equal_to< _Tp >, [2045](#)
 - std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >, [2585](#)
 - std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >, [2588](#)
 - std::exponential_distribution< _RealType >, [2055](#)
 - std::extreme_value_distribution< _RealType >, [2058](#)
 - std::fisher_f_distribution< _RealType >, [2068](#)
 - std::gamma_distribution< _RealType >, [2110](#)
 - std::geometric_distribution< _IntType >, [2113](#)
 - std::greater< _Tp >, [2126](#)
 - std::greater_equal< _Tp >, [2128](#)
 - std::hash< __gnu_cxx::throw_value_limit >, [2138](#)
 - std::hash< __gnu_cxx::throw_value_random >, [2139](#)
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, [2172](#)
 - std::less< _Tp >, [2272](#)
 - std::less_equal< _Tp >, [2274](#)
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2277](#)
 - std::logical_and< _Tp >, [2319](#)
 - std::logical_not< _Tp >, [2321](#)
 - std::logical_or< _Tp >, [2322](#)
 - std::lognormal_distribution< _RealType >, [2324](#)
 - std::mem_fun1_ref_t< _Ret, _Tp, _Arg >, [2377](#)
 - std::mem_fun1_t< _Ret, _Tp, _Arg >, [2379](#)
 - std::mem_fun_ref_t< _Ret, _Tp >, [2380](#)
 - std::mem_fun_t< _Ret, _Tp >, [2381](#)
 - std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, [2383](#)
 - std::minus< _Tp >, [2392](#)
 - std::modulus< _Tp >, [2394](#)
 - std::multiplies< _Tp >, [2455](#)
 - std::negate< _Tp >, [2485](#)
 - std::negative_binomial_distribution< _IntType >, [2486](#)
 - std::normal_distribution< _RealType >, [2492](#)
 - std::not_equal_to< _Tp >, [2496](#)
 - std::owner_less< shared_ptr< _Tp > >, [2586](#)

- std::owner_less< void >, [2587](#)
- std::owner_less< weak_ptr< _Tp > >, [2588](#)
- std::piecewise_constant_distribution< _RealType >, [2621](#)
- std::piecewise_linear_distribution< _RealType >, [2625](#)
- std::plus< _Tp >, [2628](#)
- std::pointer_to_binary_function< _Arg1, _Arg2, _Result >, [2630](#)
- std::pointer_to_unary_function< _Arg, _Result >, [2631](#)
- std::poisson_distribution< _IntType >, [2635](#)
- std::random_device, [2656](#)
- std::seed_seq, [2735](#)
- std::shuffle_order_engine< _RandomNumberEngine, __k >, [2785](#)
- std::student_t_distribution< _RealType >, [2847](#)
- std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, [2860](#)
- std::unary_function< _Arg, _Result >, [2954](#)
- std::unary_negate< _Predicate >, [2955](#)
- std::uniform_int_distribution< _IntType >, [2959](#)
- std::uniform_real_distribution< _RealType >, [2961](#)
- std::weibull_distribution< _RealType >, [3143](#)
- rethrow_exception
 - Exceptions, [138](#)
- rethrow_if_nested
 - Exceptions, [138](#)
- return_temporary_buffer
 - std, [703](#)
- reverse
 - Mutating, [35](#)
 - std::forward_list< _Tp, _Alloc >, [2088](#)
 - std::list< _Tp, _Alloc >, [2303](#)
- reverse_copy
 - Mutating, [36](#)
- reverse_iteration
 - __gnu_pbds::container_traits< Cntnr >, [1878](#)
- reverse_iterator
 - std::reverse_iterator< _Iterator >, [2709](#)
 - std::set< _Key, _Compare, _Alloc >, [2747](#)
- rfind
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [929–931](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1588](#), [1589](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1653](#), [1654](#)
- riemann_zeta
 - Mathematical Special Functions, [221](#)
 - TR1 Mathematical Special Functions, [266](#)
- riemann_zetaf
 - Mathematical Special Functions, [221](#)
- riemann_zetal
 - Mathematical Special Functions, [222](#)
- right
 - std, [703](#)
 - std::basic_fstream< _CharT, _Traits >, [1239](#)
 - std::basic_ifstream< _CharT, _Traits >, [1282](#)
 - std::basic_ios< _CharT, _Traits >, [1305](#)
 - std::basic_iostream< _CharT, _Traits >, [1351](#)
 - std::basic_istream< _CharT, _Traits >, [1389](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1430](#)
 - std::basic_ofstream< _CharT, _Traits >, [1464](#)
 - std::basic_ostream< _CharT, _Traits >, [1496](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1528](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1724](#)
 - std::ios_base, [2201](#)
- rope, [3450](#)
- ropeimpl.h, [3453](#)
- rotate
 - Mutating, [36](#)
- rotate_copy
 - Mutating, [37](#)
- rotate_fn_imps.hpp, [3405](#)
- rotr
 - Bit manipulation, [185](#)
- rotr
 - Bit manipulation, [186](#)
- round
 - Time, [421](#), [423](#)
- round_error
 - std::numeric_limits< _Tp >, [2530](#)
- round_style
 - std::__numeric_limits_base, [865](#)
 - std::numeric_limits< _Tp >, [2533](#)
- round_to_nearest
 - std, [635](#)
- round_toward_infinity
 - std, [635](#)
- round_toward_neg_infinity
 - std, [635](#)
- round_toward_zero
 - std, [635](#)
- runtime_error
 - std::runtime_error, [2718](#)
- safe_base.h, [3379](#)
- safe_container.h, [3379](#)
- safe_iterator.h, [3380](#)
- safe_iterator.tcc, [3380](#)
- safe_local_iterator.h, [3382](#)
- safe_local_iterator.tcc, [3382](#)
- safe_sequence.h, [3382](#)
- safe_sequence.tcc, [3383](#)

- safe_unordered_base.h, [3383](#)
- safe_unordered_container.h, [3383](#)
- safe_unordered_container.tcc, [3383](#)
- sample
 - std, [703](#)
 - std::experimental, [765](#)
- sample_probe_fn
 - __gnu_pbds::sample_probe_fn, [2719](#)
- sample_probe_fn.hpp, [3430](#)
- sample_range_hashing
 - __gnu_pbds::sample_range_hashing, [2720](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2727](#)
 - __gnu_pbds::sample_size_policy, [2728](#)
- sample_range_hashing.hpp, [3430](#)
- sample_ranged_hash_fn
 - __gnu_pbds::sample_ranged_hash_fn, [2721](#)
- sample_ranged_hash_fn.hpp, [3430](#)
- sample_ranged_probe_fn.hpp, [3430](#)
- sample_resize_policy
 - __gnu_pbds::sample_resize_policy, [2723](#)
- sample_resize_policy.hpp, [3438](#)
- sample_resize_trigger
 - __gnu_pbds::sample_resize_trigger, [2725](#)
- sample_resize_trigger.hpp, [3438](#)
- sample_size_policy
 - __gnu_pbds::sample_size_policy, [2728](#)
- sample_size_policy.hpp, [3438](#)
- sample_tree_node_update.hpp, [3441](#)
- sample_trie_access_traits.hpp, [3441](#)
- sample_trie_node_update
 - __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >, [2730](#)
- sample_trie_node_update.hpp, [3441](#)
- sample_update_policy
 - __gnu_pbds::sample_update_policy, [2731](#)
- sample_update_policy.hpp, [3433](#)
- sbumpc
 - __gnu_cxx::enc_filebuf< _CharT >, [2030](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2815](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2832](#)
 - std::basic_filebuf< _CharT, _Traits >, [1181](#)
 - std::basic_streambuf< _CharT, _Traits >, [1548](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1669](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3134](#)
- scan_is
 - std::__ctype_abstract_base< _CharT >, [833](#)
 - std::ctype< _CharT >, [1894](#)
 - std::ctype< char >, [1906](#)
 - std::ctype< wchar_t >, [1919](#)
 - std::ctype_byname< _CharT >, [1933](#)
 - std::ctype_byname< char >, [1944](#)
- scan_not
 - std::__ctype_abstract_base< _CharT >, [834](#)
 - std::ctype< _CharT >, [1895](#)
 - std::ctype< char >, [1906](#)
 - std::ctype< wchar_t >, [1920](#)
 - std::ctype_byname< _CharT >, [1934](#)
 - std::ctype_byname< char >, [1944](#)
- scientific
 - std, [703](#)
 - std::basic_fstream< _CharT, _Traits >, [1239](#)
 - std::basic_ifstream< _CharT, _Traits >, [1282](#)
 - std::basic_ios< _CharT, _Traits >, [1305](#)
 - std::basic_iostream< _CharT, _Traits >, [1351](#)
 - std::basic_istream< _CharT, _Traits >, [1389](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1430](#)
 - std::basic_ofstream< _CharT, _Traits >, [1464](#)
 - std::basic_ostream< _CharT, _Traits >, [1496](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1528](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1724](#)
 - std::ios_base, [2201](#)
- scoped_allocator, [3537](#)
- search
 - Non-Mutating, [59](#), [60](#)
 - std, [703](#)
 - std::__parallel, [741](#)
- search.h, [3526](#)
- search_minimal_n
 - __gnu_parallel::Settings, [1078](#)
- search_n
 - Non-Mutating, [60](#), [61](#)
- second
 - __gnu_parallel::IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [969](#)
 - std::pair< _T1, _T2 >, [2593](#)
 - std::sub_match< _Bilter >, [2857](#)
- second_argument_type
 - __gnu_cxx::project1st< _Arg1, _Arg2 >, [2645](#)
 - __gnu_cxx::project2nd< _Arg1, _Arg2 >, [2646](#)
 - __gnu_parallel::EqualFromLess< _T1, _T2, _Compare >, [950](#)
 - __gnu_parallel::EqualTo< _T1, _T2 >, [951](#)
 - __gnu_parallel::Less< _T1, _T2 >, [973](#)
 - __gnu_parallel::Lexicographic< _T1, _T2, _Compare >, [974](#)
 - __gnu_parallel::LexicographicReverse< _T1, _T2, _Compare >, [975](#)
 - __gnu_parallel::Multiplies< _Tp1, _Tp2, _Result >, [999](#)
 - __gnu_parallel::Plus< _Tp1, _Tp2, _Result >, [1014](#)
 - std::binary_function< _Arg1, _Arg2, _Result >, [1741](#)
 - std::binary_negate< _Predicate >, [1751](#)

- std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >, 1861
- std::const_mem_fun1_t< _Ret, _Tp, _Arg >, 1862
- std::divides< _Tp >, 2000
- std::equal_to< _Tp >, 2045
- std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >, 2585
- std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >, 2588
- std::greater< _Tp >, 2126
- std::greater_equal< _Tp >, 2128
- std::less< _Tp >, 2273
- std::less_equal< _Tp >, 2274
- std::logical_and< _Tp >, 2319
- std::logical_or< _Tp >, 2322
- std::mem_fun1_ref_t< _Ret, _Tp, _Arg >, 2378
- std::mem_fun1_t< _Ret, _Tp, _Arg >, 2379
- std::minus< _Tp >, 2392
- std::modulus< _Tp >, 2395
- std::multiplies< _Tp >, 2455
- std::not_equal_to< _Tp >, 2496
- std::owner_less< shared_ptr< _Tp > >, 2586
- std::owner_less< void >, 2587
- std::owner_less< weak_ptr< _Tp > >, 2588
- std::plus< _Tp >, 2628
- std::pointer_to_binary_function< _Arg1, _Arg2, _Result >, 2630
- second_type
 - __gnu_parallel::IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, 967
 - std::pair< _T1, _T2 >, 2592
 - std::sub_match< _Bilter >, 2853
- seconds
 - Time, 413
- seed
 - std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1994, 1995
 - std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2174, 2175
 - std::linear_congruential_engine< _UIntType, __a, __c, __m >, 2279
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, 2787, 2788
 - std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2861
- seed_seq
 - std::seed_seq, 2735
- seekdir
 - std::basic_fstream< _CharT, _Traits >, 1199
 - std::basic_ifstream< _CharT, _Traits >, 1249
 - std::basic_ios< _CharT, _Traits >, 1290
 - std::basic_iostream< _CharT, _Traits >, 1314
 - std::basic_istream< _CharT, _Traits >, 1359
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1396
- std::basic_ofstream< _CharT, _Traits >, 1438
- std::basic_ostream< _CharT, _Traits >, 1471
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1503
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1686
- std::ios_base, 2191
- seekg
 - std::basic_fstream< _CharT, _Traits >, 1228, 1229
 - std::basic_ifstream< _CharT, _Traits >, 1273
 - std::basic_iostream< _CharT, _Traits >, 1340, 1341
 - std::basic_istream< _CharT, _Traits >, 1380
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1419, 1420
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1713, 1714
- seekoff
 - __gnu_cxx::enc_filebuf< _CharT >, 2031
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2816
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2832
 - std::basic_filebuf< _CharT, _Traits >, 1181
 - std::basic_streambuf< _CharT, _Traits >, 1548
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1669
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3134
- seekp
 - std::basic_fstream< _CharT, _Traits >, 1229, 1230
 - std::basic_iostream< _CharT, _Traits >, 1341, 1342
 - std::basic_ofstream< _CharT, _Traits >, 1456
 - std::basic_ostream< _CharT, _Traits >, 1487, 1488
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1518, 1519
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1714, 1715
- seekpos
 - __gnu_cxx::enc_filebuf< _CharT >, 2031
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2816
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2833
 - std::basic_filebuf< _CharT, _Traits >, 1182
 - std::basic_streambuf< _CharT, _Traits >, 1549
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1669
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3134
- select_on_container_copy_construction
 - __gnu_cxx::__alloc_traits< _Alloc, typename >, 810
 - std::allocator_traits< _Alloc >, 1097
 - std::allocator_traits< allocator< _Tp > >, 1102
 - std::allocator_traits< allocator< void > >, 1105
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, 1110
- semaphore, 3537
- semaphore_base.h, 3261
- sentry

- std::basic_istream< _CharT, _Traits >::sentry, [2738](#)
- std::basic_ostream< _CharT, _Traits >::sentry, [2739](#)
- Sequences, [129](#)
- sequential
 - __gnu_parallel, [468](#)
- set, [3537](#), [3538](#)
 - __gnu_parallel::Settings, [1074](#)
 - std::bitset< _Nb >, [1773](#), [1774](#)
 - std::set< _Key, _Compare, _Alloc >, [2748–2751](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2016](#)
- Set Operations, [97](#)
 - includes, [98](#), [99](#)
 - set_difference, [99](#), [100](#)
 - set_intersection, [101](#)
 - set_symmetric_difference, [102](#)
 - set_union, [103](#), [104](#)
- set.h, [3384](#)
- set_default_resource
 - experimental/memory_resource, [3484](#)
 - memory_resource, [3485](#)
- set_difference
 - Set Operations, [99](#), [100](#)
- set_difference_minimal_n
 - __gnu_parallel::Settings, [1078](#)
- set_intersection
 - Set Operations, [101](#)
- set_intersection_minimal_n
 - __gnu_parallel::Settings, [1078](#)
- set_load
 - __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >, [1784](#)
- set_loads
 - __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >, [2157](#)
 - __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >, [2168](#)
- set_new_handler
 - std, [704](#)
- set_num_threads
 - __gnu_parallel::balanced_quicksort_tag, [1167](#)
 - __gnu_parallel::balanced_tag, [1168](#)
 - __gnu_parallel::default_parallel_tag, [1958](#)
 - __gnu_parallel::exact_tag, [2051](#)
 - __gnu_parallel::multiway_mergesort_exact_tag, [2481](#)
 - __gnu_parallel::multiway_mergesort_sampling_tag, [2482](#)
 - __gnu_parallel::multiway_mergesort_tag, [2483](#)
 - __gnu_parallel::omp_loop_static_tag, [2562](#)
 - __gnu_parallel::omp_loop_tag, [2563](#)
 - __gnu_parallel::parallel_tag, [2598](#)
 - __gnu_parallel::quicksort_tag, [2654](#)
 - __gnu_parallel::sampling_tag, [2732](#)
 - __gnu_parallel::unbalanced_tag, [2956](#)
- set_operations.h, [3526](#)
- set_symmetric_difference
 - Set Operations, [102](#)
- set_symmetric_difference_minimal_n
 - __gnu_parallel::Settings, [1078](#)
- set_terminate
 - Exceptions, [138](#)
- set_unexpected
 - Exceptions, [138](#)
- set_union
 - Set Operations, [103](#), [104](#)
- set_union_minimal_n
 - __gnu_parallel::Settings, [1078](#)
- setbase
 - std, [704](#)
- setbuf
 - __gnu_cxx::enc_filebuf< _CharT >, [2031](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2816](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2833](#)
 - std::basic_filebuf< _CharT, _Traits >, [1182](#)
 - std::basic_streambuf< _CharT, _Traits >, [1549](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1669](#)
 - std::wbuffer_convert< _Codecvt, Elem, _Tr >, [3134](#)
- setf
 - std::basic_fstream< _CharT, _Traits >, [1230](#)
 - std::basic_ifstream< _CharT, _Traits >, [1274](#)
 - std::basic_ios< _CharT, _Traits >, [1298](#), [1299](#)
 - std::basic_iostream< _CharT, _Traits >, [1342](#), [1343](#)
 - std::basic_istream< _CharT, _Traits >, [1381](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1420](#), [1422](#)
 - std::basic_ofstream< _CharT, _Traits >, [1456](#), [1457](#)
 - std::basic_ostream< _CharT, _Traits >, [1488](#), [1489](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1519](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1715](#)
 - std::ios_base, [2194](#), [2196](#)
- setfill
 - std, [704](#)
- setg
 - __gnu_cxx::enc_filebuf< _CharT >, [2032](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2817](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2833](#)
 - std::basic_filebuf< _CharT, _Traits >, [1182](#)
 - std::basic_streambuf< _CharT, _Traits >, [1549](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1670](#)
 - std::wbuffer_convert< _Codecvt, Elem, _Tr >, [3135](#)
- setiosflags
 - std, [704](#)
- setp

- __gnu_cxx::enc_filebuf< _CharT >, 2032
- __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2817
- __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2834
- std::basic_filebuf< _CharT, _Traits >, 1183
- std::basic_streambuf< _CharT, _Traits >, 1550
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1670
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3135
- setprecision
 - std, 705
- setstate
 - std::basic_fstream< _CharT, _Traits >, 1231
 - std::basic_ifstream< _CharT, _Traits >, 1274
 - std::basic_ios< _CharT, _Traits >, 1299
 - std::basic_iostream< _CharT, _Traits >, 1343
 - std::basic_istream< _CharT, _Traits >, 1381
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1422
 - std::basic_ofstream< _CharT, _Traits >, 1457
 - std::basic_ostream< _CharT, _Traits >, 1489
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1521
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1716
- settings.h, 3527
 - _GLIBCXX_PARALLEL_CONDITION, 3527
- setw
 - std, 705
- sgetc
 - __gnu_cxx::enc_filebuf< _CharT >, 2033
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2818
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2834
 - std::basic_filebuf< _CharT, _Traits >, 1183
 - std::basic_streambuf< _CharT, _Traits >, 1550
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1671
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3136
- sgetn
 - __gnu_cxx::enc_filebuf< _CharT >, 2033
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2818
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2834
 - std::basic_filebuf< _CharT, _Traits >, 1183
 - std::basic_streambuf< _CharT, _Traits >, 1550
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1671
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3136
- SGL, 155
 - _Find_first, 158
 - _Find_next, 158
 - _Unchecked_flip, 159
 - _Unchecked_reset, 159
 - _Unchecked_set, 159
 - _Unchecked_test, 159
 - _median, 157, 158
- compose1, 159
- compose2, 159
- constant0, 160
- constant1, 160
- constant2, 160
- copy_n, 160
- distance, 160
- identity_element, 161
- lexicographical_compare_3way, 161
- power, 161, 162
- random_sample, 162
- random_sample_n, 162
- uninitialized_copy_n, 163
- shared_future
 - std::shared_future< _Res >, 2767
 - std::shared_future< _Res & >, 2769
 - std::shared_future< void >, 2770, 2771
- shared_mutex, 3539
- shared_ptr
 - std::shared_ptr< _Tp >, 2777–2782
- shared_ptr.h, 3261, 3262
- shared_ptr_atomic.h, 3264
- shared_ptr_base.h, 3265
- shift
 - Numeric Arrays, 240
- showbase
 - std, 705
 - std::basic_fstream< _CharT, _Traits >, 1239
 - std::basic_ifstream< _CharT, _Traits >, 1282
 - std::basic_ios< _CharT, _Traits >, 1305
 - std::basic_iostream< _CharT, _Traits >, 1351
 - std::basic_istream< _CharT, _Traits >, 1389
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1430
 - std::basic_ofstream< _CharT, _Traits >, 1464
 - std::basic_ostream< _CharT, _Traits >, 1496
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1528
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1724
 - std::ios_base, 2201
- showmanyc
 - __gnu_cxx::enc_filebuf< _CharT >, 2033
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2818
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2835
 - std::basic_filebuf< _CharT, _Traits >, 1184
 - std::basic_streambuf< _CharT, _Traits >, 1551
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1671
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3136
- showpoint
 - std, 705
 - std::basic_fstream< _CharT, _Traits >, 1239
 - std::basic_ifstream< _CharT, _Traits >, 1282

- std::basic_ios< _CharT, _Traits >, 1305
- std::basic_iostream< _CharT, _Traits >, 1351
- std::basic_istream< _CharT, _Traits >, 1389
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1430
- std::basic_ofstream< _CharT, _Traits >, 1464
- std::basic_ostream< _CharT, _Traits >, 1496
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1528
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1724
- std::ios_base, 2201
- showpos
 - std, 705
 - std::basic_fstream< _CharT, _Traits >, 1239
 - std::basic_ifstream< _CharT, _Traits >, 1282
 - std::basic_ios< _CharT, _Traits >, 1306
 - std::basic_iostream< _CharT, _Traits >, 1351
 - std::basic_istream< _CharT, _Traits >, 1389
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1430
 - std::basic_ofstream< _CharT, _Traits >, 1464
 - std::basic_ostream< _CharT, _Traits >, 1496
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1528
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1724
 - std::ios_base, 2201
- shrink_to_fit
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 931
 - std::basic_string< _CharT, _Traits, _Alloc >, 1655
 - std::deque< _Tp, _Alloc >, 1986
 - std::vector< _Tp, _Alloc >, 3120
- shuffle
 - Mutating, 37
- shuffle_order_engine
 - std::shuffle_order_engine< _RandomNumberEngine, __k >, 2785, 2786
- signaling_NaN
 - std::numeric_limits< _Tp >, 2531
- simd, 3398
- simd_abi::deduce< _Tp, _Np,... >, 1954
- sin
 - Complex Numbers, 195
- sinh
 - Complex Numbers, 195
- size
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 931
 - __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, 2869
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, 1589
 - Numeric Arrays, 240, 241
 - std, 706
 - std::Temporary_buffer< _ForwardIterator, _Tp >, 1083
 - std::basic_string< _CharT, _Traits, _Alloc >, 1655
 - std::bitset< _Nb >, 1774
 - std::deque< _Tp, _Alloc >, 1986
 - std::list< _Tp, _Alloc >, 2303
 - std::map< _Key, _Tp, _Compare, _Alloc >, 2361
 - std::match_results< _Bi_iter, _Alloc >, 2375
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, 2452
 - std::multiset< _Key, _Compare, _Alloc >, 2477
 - std::priority_queue< _Tp, _Sequence, _Compare >, 2643
 - std::queue< _Tp, _Sequence >, 2653
 - std::set< _Key, _Compare, _Alloc >, 2764
 - std::stack< _Tp, _Sequence >, 2803
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2016
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 3007
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3036
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3063
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3090
 - std::vector< _Tp, _Alloc >, 3120
- size_fn_imps.hpp, 3425
- size_type
 - __gnu_pbds::hash_prime_size_policy, 2163
 - __gnu_pbds::sample_range_hashing, 2720
 - __gnu_pbds::sample_resize_policy, 2722
 - __gnu_pbds::sample_resize_trigger, 2725
 - __gnu_pbds::sample_size_policy, 2728
 - __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, ATraits, _Alloc >, 2934
- std::allocator_traits< _Alloc >, 1095
- std::allocator_traits< allocator< _Tp > >, 1100
- std::allocator_traits< allocator< void > >, 1104
- std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, 1108
- std::set< _Key, _Compare, _Alloc >, 2748
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 2985
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 3017
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 3044
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3071
- skipws
 - std, 706
 - std::basic_fstream< _CharT, _Traits >, 1239

- std::basic_ifstream< _CharT, _Traits >, 1283
- std::basic_ios< _CharT, _Traits >, 1306
- std::basic_iostream< _CharT, _Traits >, 1352
- std::basic_istream< _CharT, _Traits >, 1389
- std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1431
- std::basic_ofstream< _CharT, _Traits >, 1464
- std::basic_ostream< _CharT, _Traits >, 1496
- std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1529
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1725
- std::ios_base, 2201
- sleep_for
 - std::this_thread, 786
- sleep_until
 - std::this_thread, 786
- slice
 - Numeric Arrays, 228
- slice_array
 - Numeric Arrays, 229
- slice_array.h, 3266
- slist, 3454
- snnextc
 - __gnu_cxx::enc_filebuf< _CharT >, 2034
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2819
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2835
 - std::basic_filebuf< _CharT, _Traits >, 1184
 - std::basic_streambuf< _CharT, _Traits >, 1551
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1672
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3137
- sort
 - Sorting, 84
 - std::forward_list< _Tp, _Alloc >, 2088, 2089
 - std::list< _Tp, _Alloc >, 2303
- sort.h, 3528
- sort_heap
 - Heap, 96, 97
- sort_minimal_n
 - __gnu_parallel:: Settings, 1078
- sort_mwms_oversampling
 - __gnu_parallel:: Settings, 1078
- sort_qs_num_samples_preset
 - __gnu_parallel:: Settings, 1078
- sort_qsb_base_case_maximal_n
 - __gnu_parallel:: Settings, 1078
- Sorting, 62
 - clamp, 64, 65
 - inplace_merge, 65, 66
 - is_sorted, 67
 - is_sorted_until, 67, 68
 - lexicographical_compare, 68, 69
 - max, 69, 71
 - max_element, 71, 72
 - merge, 72, 73
 - min, 73, 74
 - min_element, 75
 - minmax, 76
 - minmax_element, 76, 77
 - next_permutation, 77, 79
 - nth_element, 79, 80
 - partial_sort, 80, 81
 - partial_sort_copy, 82
 - prev_permutation, 83
 - sort, 84
 - stable_sort, 85
- source_location, 3539
- span, 3539
- specfun.h, 3267
- sph_bessel
 - Mathematical Special Functions, 222
 - TR1 Mathematical Special Functions, 266
- sph_besself
 - Mathematical Special Functions, 222
- sph_bessell
 - Mathematical Special Functions, 223
- sph_legendre
 - Mathematical Special Functions, 223
 - TR1 Mathematical Special Functions, 266
- sph_legendref
 - Mathematical Special Functions, 223
- sph_legendrel
 - Mathematical Special Functions, 223
- sph_neumann
 - Mathematical Special Functions, 224
 - TR1 Mathematical Special Functions, 266
- sph_neumannf
 - Mathematical Special Functions, 224
- sph_neumannl
 - Mathematical Special Functions, 224
- splay_fn_imps.hpp, 3438
- splay_tree_.hpp, 3438
- splice
 - std::list< _Tp, _Alloc >, 2303–2305
- splice_after
 - std::forward_list< _Tp, _Alloc >, 2089, 2090
- split_fn_imps.hpp, 3435
- split_join_can_throw
 - __gnu_pbds::container_traits< Cntr >, 1878
- split_join_fn_imps.hpp, 3419
- sputbackc
 - __gnu_cxx::enc_filebuf< _CharT >, 2034
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2819
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2835
 - std::basic_filebuf< _CharT, _Traits >, 1184
 - std::basic_streambuf< _CharT, _Traits >, 1551

- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1672
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3137
- sputc
 - __gnu_cxx::enc_filebuf< _CharT >, 2034
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2819
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2836
 - std::basic_filebuf< _CharT, _Traits >, 1185
 - std::basic_streambuf< _CharT, _Traits >, 1552
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1672
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3137
- sputn
 - __gnu_cxx::enc_filebuf< _CharT >, 2035
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2820
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2836
 - std::basic_filebuf< _CharT, _Traits >, 1185
 - std::basic_streambuf< _CharT, _Traits >, 1552
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1673
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3138
- sqrt
 - Complex Numbers, 195
- sqrt2_v
 - std::numbers, 773
- sqrt3_v
 - std::numbers, 773
- sregex_token_iterator
 - Regular Expressions, 272
- sso_string_base.h, 3455
- sstream, 3540
- sstream.tcc, 3269
- ssub_match
 - Regular Expressions, 272
- stable_partition
 - Mutating, 38
- stable_sort
 - Sorting, 85
- stack, 3541
 - std::stack< _Tp, _Sequence >, 2803
- standard_policies.hpp, 3439
- start
 - Numeric Arrays, 241
- state
 - std::fpos< _StateT >, 2093
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3138
 - std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >, 3147
- static_pointer_cast
 - Pointer Abstractions, 385, 386
- std, 510
 - _Construct, 643
 - _Destroy, 643, 644
 - _Destroy_n, 644
 - __final_insertion_sort, 635
 - __find_if, 635
 - __find_if_not, 635
 - __find_if_not_n, 636
 - __gcd, 636
 - __gen_two_uniform_ints, 636
 - __heap_select, 636
 - __inplace_stable_sort, 637
 - __insertion_sort, 637
 - __introsort_loop, 637
 - __joinit, 710
 - __lg, 637
 - __merge_adaptive, 637
 - __merge_without_buffer, 638
 - __move_median_to_first, 638
 - __move_merge, 638
 - __move_merge_adaptive, 638
 - __move_merge_adaptive_backward, 639
 - __partition, 639
 - __ptr_rebind, 632
 - __reverse, 639
 - __rotate, 640
 - __rotate_adaptive, 640
 - __sample, 641
 - __search_n_aux, 641
 - __stable_partition_adaptive, 641
 - __umap_traits, 632
 - __ummap_traits, 632
 - __umset_traits, 632
 - __unguarded_insertion_sort, 642
 - __unguarded_linear_insert, 642
 - __unguarded_partition, 642
 - __unguarded_partition_pivot, 642
 - __unique_copy, 643
 - __uset_traits, 633
 - acos, 644
 - acosh, 644
 - advance, 644
 - arg, 645
 - asin, 645
 - asinh, 645
 - atan, 645
 - atanh, 645
 - begin, 646
 - bind_front, 646
 - boolalpha, 647
 - byte, 634
 - cbegin, 647
 - cend, 647
 - cerr, 710
 - chars_format, 634
 - cin, 710
 - clog, 710
 - compare_three_way_result_t, 633

const_pointer_cast, 647
cout, 710
crbegin, 647
crend, 648
data, 648
dec, 649
default_sentinel, 711
defaultfloat, 649
denorm_absent, 634
denorm_indeterminate, 634
denorm_present, 634
destroying_delete, 711
distance, 649
dynamic_pointer_cast, 649
empty, 650
end, 650, 651
endl, 651
ends, 651
exchange, 651
fabs, 651
fixed, 651
float_denorm_style, 634
float_round_style, 635
flush, 652
from_chars, 652
gcd, 652
get, 652–654
get_money, 654
get_new_handler, 654
get_temporary_buffer, 654
get_time, 655
getline, 655–657
hex, 657
hexfloat, 657
index_sequence, 633
index_sequence_for, 633
internal, 657
invoke, 657
io_errc, 635
isalnum, 658
isalpha, 658
isblank, 658
iscntrl, 658
isdigit, 658
isgraph, 658
islower, 659
isprint, 659
ispunct, 659
isspace, 659
isupper, 659
isxdigit, 659
launder, 659
lcm, 660
left, 660
make_index_sequence, 633
make_integer_sequence, 633
new_handler, 633
noboolalpha, 660
noshowbase, 660
noshowpoint, 660
noshowpos, 660
noskipws, 660
nounitbuf, 660
nouppercase, 661
oct, 661
operator!=, 661–663
operator<, 665–670
operator<<, 671–681
operator<=, 681–683
operator>, 689–691
operator>>, 694–698
operator>=, 691–693
operator^, 698
operator+, 664, 665
operator==, 683–688
operator&, 663
operator|, 699
put_money, 699
put_time, 700
quoted, 700
rbegin, 700, 701
rend, 701, 702
replace_copy, 702
resetiosflags, 702
return_temporary_buffer, 703
right, 703
round_to_nearest, 635
round_toward_infinity, 635
round_toward_neg_infinity, 635
round_toward_zero, 635
sample, 703
scientific, 703
search, 703
set_new_handler, 704
setbase, 704
setfill, 704
setiosflags, 704
setprecision, 705
setw, 705
showbase, 705
showpoint, 705
showpos, 705
size, 706
skipws, 706
static_pointer_cast, 706
streamoff, 633
streampos, 633
streamsize, 634

- swap, [706–709](#)
- tolower, [709](#)
- toupper, [709](#)
- u16streampos, [634](#)
- u32streampos, [634](#)
- unitbuf, [709](#)
- uppercase, [710](#)
- wcerr, [711](#)
- wcin, [711](#)
- wclog, [711](#)
- wcout, [711](#)
- ws, [710](#)
- wstreampos, [634](#)
- std::__atomic_base<_ITp>, [810](#)
- std::__atomic_base<_PTp*>, [812](#)
- std::__atomic_flag_base, [813](#)
- std::__basic_future<_Res>, [814](#)
- _M_get_result, [815](#)
- std::__codecvt_abstract_base< _InternT, _ExternT, _StateT>, [817](#)
- do_out, [818](#)
- in, [819](#)
- out, [820](#)
- unshift, [820](#)
- std::__ctype_abstract_base<_CharT>, [824](#)
- char_type, [826](#)
- do_is, [826](#)
- do_narrow, [827](#)
- do_scan_is, [828](#)
- do_scan_not, [828](#)
- do_tolower, [829](#)
- do_toupper, [830](#)
- do_widen, [831](#)
- is, [832](#)
- narrow, [832](#), [833](#)
- scan_is, [833](#)
- scan_not, [834](#)
- tolower, [834](#), [835](#)
- toupper, [835](#), [836](#)
- widen, [836](#)
- std::__debug, [711](#)
- operator<=, [719](#)
- operator>, [719](#)
- operator>=, [719](#)
- swap, [720](#)
- std::__debug::bitset<_Nb>, [1764](#)
- std::__debug::deque<_Tp, _Allocator>, [1961](#)
- std::__debug::forward_list<_Tp, _Alloc>, [2071](#)
- std::__debug::list<_Tp, _Allocator>, [2282](#)
- std::__debug::map<_Key, _Tp, _Compare, _Allocator>, [2333](#)
- std::__debug::multimap<_Key, _Tp, _Compare, _Allocator>, [2427](#)
- std::__debug::multiset<_Key, _Compare, _Allocator>, [2456](#)
- std::__debug::set<_Key, _Compare, _Allocator>, [2740](#)
- std::__debug::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>, [2975](#)
- std::__debug::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>, [3009](#)
- std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc>, [3036](#)
- std::__debug::unordered_set<_Value, _Hash, _Pred, _Alloc>, [3063](#)
- std::__debug::vector<_Tp, _Allocator>, [3099](#)
- vector, [3102](#)
- std::__detail, [720](#)
- __from_chars_alnum, [723](#)
- __from_chars_pow2_base, [723](#)
- operator<<, [723](#)
- operator>>, [724](#)
- std::__detail::BracketMatcher<_TraitsT, __icase, __colate>, [937](#)
- std::__detail::Compiler<_TraitsT>, [940](#)
- std::__detail::Executor<_Bilter, _Alloc, _TraitsT, __dfs_mode>, [951](#)
- std::__detail::List_node_base, [979](#)
- std::__detail::List_node_header, [980](#)
- std::__detail::Quoted_string<_String, _CharT>, [1021](#)
- std::__detail::Scanner<_CharT>, [1071](#)
- _TokenT, [1072](#)
- std::__detail::StateSeq<_TraitsT>, [1081](#)
- std::__exception_ptr::exception_ptr, [2053](#)
- std::__is_location_invariant<_Tp>, [852](#)
- std::__is_nullptr_t<_Tp>, [853](#)
- std::__new_allocator<_Tp>, [861](#)
- std::__numeric_limits_base, [862](#)
- digits, [863](#)
- digits10, [863](#)
- has_denorm, [863](#)
- has_denorm_loss, [863](#)
- has_infinity, [864](#)
- has_quiet_NaN, [864](#)
- has_signaling_NaN, [864](#)
- is_bounded, [864](#)
- is_exact, [864](#)
- is_iec559, [864](#)
- is_integer, [864](#)
- is_modulo, [864](#)
- is_signed, [864](#)
- is_specialized, [865](#)
- max_digits10, [865](#)
- max_exponent, [865](#)
- max_exponent10, [865](#)
- min_exponent, [865](#)
- min_exponent10, [865](#)
- radix, [865](#)

round_style, 865
 tinyness_before, 866
 traps, 866
 std::__parallel, 724
 search, 741
 std::__parallel::__CRandNumber<_MustBeInt>, 941
 std::__Base_bitset<0>, 934
 std::__Base_bitset<1>, 935
 std::__Base_bitset<_Nw>, 933
 _M_w, 934
 std::__Bind<_Signature>, 936
 std::__Bind_result<_Result, _Signature>, 937
 std::__Deque_base<_Tp, _Alloc>, 941
 _M_initialize_map, 943
 std::__Deque_iterator<_Tp, _Ref, _Ptr>, 944
 _M_set_node, 945
 std::__Function_base, 954
 std::__Fwd_list_base<_Tp, _Alloc>, 954
 std::__Fwd_list_const_iterator<_Tp>, 956
 operator==, 957
 std::__Fwd_list_iterator<_Tp>, 957
 operator==, 958
 std::__Fwd_list_node<_Tp>, 958
 std::__Fwd_list_node_base, 959
 std::__List_base<_Tp, _Alloc>, 975
 std::__List_const_iterator<_Tp>, 977
 std::__List_iterator<_Tp>, 978
 std::__List_node<_Tp>, 979
 std::__Node_handle<_Key, _Value, _NodeAlloc>, 1003
 std::__Node_handle<_Value, _Value, _NodeAlloc>, 1005
 std::__Node_handle_common<_Val, _NodeAlloc>, 1006
 std::__Node_insert_return<_Iterator, _NodeHandle>, 1006
 std::__Not_fn<_Fn>, 1010
 std::__Optional_base<_Tp, bool, bool>, 1011
 std::__Placeholder<_Num>, 1013
 std::__Sp_ebo_helper<_Nm, _Tp, false>, 1079
 std::__Sp_ebo_helper<_Nm, _Tp, true>, 1079
 std::__Temporary_buffer<_ForwardIterator, _Tp>, 1082
 _Temporary_buffer, 1083
 begin, 1083
 end, 1083
 requested_size, 1083
 size, 1083
 std::__Tuple_impl<_Idx, _Elements>, 1083
 std::__Tuple_impl<_Idx, _Head, _Tail...>, 1084
 std::__V2::condition_variable_any, 1858
 std::__V2::error_category, 2046
 default_error_condition, 2047
 equivalent, 2047
 name, 2047
 operator<, 2047
 operator==, 2047
 std::__Vector_base<_Tp, _Alloc>, 1085
 std::add_const<_Tp>, 1087
 std::add_cv<_Tp>, 1087
 std::add_lvalue_reference<_Tp>, 1088
 std::add_pointer<_Tp>, 1088
 std::add_rvalue_reference<_Tp>, 1088
 std::add_volatile<_Tp>, 1089
 std::adopt_lock_t, 1089
 std::aligned_storage<_Len, _Align>, 1089
 std::aligned_union<_Len, _Types>, 1089
 type, 1090
 std::alignment_of<_Tp>, 1090
 std::allocator<_Tp>, 1091
 std::allocator<void>, 1092
 std::allocator_traits<_Alloc>, 1092
 allocate, 1095, 1096
 allocator_type, 1094
 const_pointer, 1094
 const_void_pointer, 1094
 construct, 1096
 deallocate, 1096
 destroy, 1097
 difference_type, 1094
 is_always_equal, 1094
 max_size, 1097
 pointer, 1094
 propagate_on_container_copy_assignment, 1095
 propagate_on_container_move_assignment, 1095
 propagate_on_container_swap, 1095
 select_on_container_copy_construction, 1097
 size_type, 1095
 value_type, 1095
 void_pointer, 1095
 std::allocator_traits<allocator<_Tp>>, 1098
 allocate, 1100
 allocator_type, 1099
 const_pointer, 1099
 const_void_pointer, 1099
 construct, 1101
 deallocate, 1101
 destroy, 1101
 difference_type, 1099
 is_always_equal, 1099
 max_size, 1102
 pointer, 1099
 propagate_on_container_copy_assignment, 1099
 propagate_on_container_move_assignment, 1099
 propagate_on_container_swap, 1099
 select_on_container_copy_construction, 1102
 size_type, 1100
 value_type, 1100
 void_pointer, 1100
 std::allocator_traits<allocator<void>>, 1102
 allocate, 1104
 allocator_type, 1103

- const_pointer, 1103
- const_void_pointer, 1103
- construct, 1104
- deallocate, 1105
- destroy, 1105
- difference_type, 1103
- is_always_equal, 1104
- max_size, 1105
- pointer, 1104
- propagate_on_container_copy_assignment, 1104
- propagate_on_container_move_assignment, 1104
- propagate_on_container_swap, 1104
- select_on_container_copy_construction, 1105
- size_type, 1104
- value_type, 1104
- void_pointer, 1104
- std::allocator_traits< pmr::polymorphic_allocator< _Tp >
>, 1106
 - allocate, 1108
 - allocator_type, 1107
 - const_pointer, 1107
 - const_void_pointer, 1107
 - construct, 1109
 - deallocate, 1109
 - destroy, 1109
 - difference_type, 1107
 - is_always_equal, 1107
 - max_size, 1110
 - pointer, 1107
 - propagate_on_container_copy_assignment, 1107
 - propagate_on_container_move_assignment, 1107
 - propagate_on_container_swap, 1107
 - select_on_container_copy_construction, 1110
 - size_type, 1108
 - value_type, 1108
 - void_pointer, 1108
- std::any, 1112
 - ~any, 1113
 - any, 1112, 1113
 - emplace, 1113
 - has_value, 1114
 - operator=, 1114
 - reset, 1114
 - swap, 1114
 - type, 1114
- std::array< _Tp, _Nm >, 1117
- std::assignable_from, 791
- std::atomic< _Tp >, 1119
- std::atomic< _Tp * >, 1120
- std::atomic< bool >, 1121
- std::atomic< char >, 1122
- std::atomic< char16_t >, 1124
- std::atomic< char32_t >, 1126
- std::atomic< int >, 1128
- std::atomic< long >, 1130
- std::atomic< long long >, 1132
- std::atomic< short >, 1134
- std::atomic< signed char >, 1136
- std::atomic< unsigned char >, 1138
- std::atomic< unsigned int >, 1140
- std::atomic< unsigned long >, 1142
- std::atomic< unsigned long long >, 1144
- std::atomic< unsigned short >, 1146
- std::atomic< wchar_t >, 1148
- std::atomic_flag, 1150
- std::atomic_ref< _Tp >, 1151
- std::auto_ptr< _Tp >, 1151
 - ~auto_ptr, 1153
 - auto_ptr, 1152, 1153
 - element_type, 1152
 - get, 1154
 - operator*, 1154
 - operator->, 1154
 - operator=, 1154
 - release, 1155
 - reset, 1155
- std::auto_ptr_ref< _Tp1 >, 1155
- std::back_insert_iterator< _Container >, 1156
 - back_insert_iterator, 1157
 - container_type, 1157
 - iterator_category, 1157
 - operator*, 1157
 - operator++, 1158
 - operator=, 1158
 - pointer, 1157
 - reference, 1157
 - value_type, 1157
- std::bad_alloc, 1158
 - what, 1159
- std::bad_any_cast, 1159
 - what, 1160
- std::bad_cast, 1161
 - what, 1161
- std::bad_exception, 1162
 - what, 1162
- std::bad_function_call, 1162
 - what, 1163
- std::bad_optional_access, 1163
 - what, 1164
- std::bad_typeid, 1165
 - what, 1165
- std::bad_weak_ptr, 1165
 - what, 1166
- std::basic_filebuf< _CharT, _Traits >, 1170
 - _M_buf, 1188
 - _M_buf_locale, 1188
 - _M_buf_size, 1188
 - _M_create_pback, 1174

[_M_destroy_pback](#), 1174
[_M_ext_buf](#), 1188
[_M_ext_buf_size](#), 1188
[_M_ext_next](#), 1188
[_M_in_beg](#), 1188
[_M_in_cur](#), 1188
[_M_in_end](#), 1189
[_M_mode](#), 1189
[_M_out_beg](#), 1189
[_M_out_cur](#), 1189
[_M_out_end](#), 1189
[_M_pback](#), 1189
[_M_pback_cur_save](#), 1189
[_M_pback_end_save](#), 1189
[_M_pback_init](#), 1190
[_M_reading](#), 1190
[_M_set_buffer](#), 1174
[~basic_filebuf](#), 1174
[basic_filebuf](#), 1173
[close](#), 1174
[eback](#), 1174
[egptr](#), 1175
[eptr](#), 1175
[gbump](#), 1175
[getloc](#), 1175
[gptr](#), 1176
[imbue](#), 1176
[in_avail](#), 1176
[is_open](#), 1176
[open](#), 1177, 1178
[overflow](#), 1178
[pbackfail](#), 1179
[pbase](#), 1179
[pbump](#), 1179
[pptr](#), 1180
[pubimbue](#), 1180
[pubseekoff](#), 1180
[pubseekpos](#), 1180
[pubsetbuf](#), 1181
[pubsync](#), 1181
[sbumpc](#), 1181
[seekoff](#), 1181
[seekpos](#), 1182
[setbuf](#), 1182
[setg](#), 1182
[setp](#), 1183
[sgetc](#), 1183
[sgetn](#), 1183
[showmanyc](#), 1184
[snextc](#), 1184
[sputbackc](#), 1184
[sputc](#), 1185
[sputn](#), 1185
[sungetc](#), 1185
[sync](#), 1186
[uflow](#), 1186
[underflow](#), 1186
[xsgetn](#), 1187
[xsputn](#), 1187
[std::basic_fstream<_CharT, _Traits >](#), 1191
[_M_gcount](#), 1235
[_M_getloc](#), 1200
[__num_put_type](#), 1197
[~basic_fstream](#), 1200
[adjustfield](#), 1235
[app](#), 1235
[ate](#), 1236
[bad](#), 1201
[badbit](#), 1236
[basefield](#), 1236
[basic_fstream](#), 1199, 1200
[beg](#), 1236
[binary](#), 1236
[boolalpha](#), 1236
[clear](#), 1201
[close](#), 1201
[copyfmt](#), 1201
[cur](#), 1236
[dec](#), 1237
[end](#), 1237
[eof](#), 1202
[eofbit](#), 1237
[event](#), 1199
[event_callback](#), 1197
[exceptions](#), 1202
[fail](#), 1203
[failbit](#), 1237
[fill](#), 1203
[fixed](#), 1237
[flags](#), 1203, 1204
[floatfield](#), 1237
[flush](#), 1204
[fmtflags](#), 1197
[gcount](#), 1204
[get](#), 1204–1206
[getline](#), 1207, 1208
[getloc](#), 1208
[good](#), 1208
[goodbit](#), 1238
[hex](#), 1238
[ignore](#), 1208, 1209
[imbue](#), 1209
[in](#), 1238
[init](#), 1210
[internal](#), 1238
[iostate](#), 1198
[is_open](#), 1210
[iword](#), 1210

- left, 1238
- narrow, 1211
- oct, 1238
- open, 1211, 1212
- openmode, 1198
- operator bool, 1212
- operator!, 1212
- operator<<, 1212–1217
- operator>>, 1218–1220, 1222–1224
- out, 1239
- peek, 1224
- precision, 1224
- put, 1225
- putback, 1225
- pword, 1226
- rdbuf, 1226
- rdstate, 1227
- read, 1227
- readsome, 1227
- register_callback, 1228
- right, 1239
- scientific, 1239
- seekdir, 1199
- seekg, 1228, 1229
- seekp, 1229, 1230
- setf, 1230
- setstate, 1231
- showbase, 1239
- showpoint, 1239
- showpos, 1239
- skipws, 1239
- sync, 1231
- sync_with_stdio, 1231
- tellg, 1232
- tellp, 1232
- tie, 1232
- trunc, 1239
- unget, 1233
- unitbuf, 1239
- unsetf, 1233
- uppercase, 1240
- widen, 1233
- width, 1234
- write, 1234
- xalloc, 1235
- std::basic_ifstream<_CharT, _Traits >, 1243
 - _M_gcount, 1278
 - _M_getloc, 1251
 - __num_put_type, 1248
 - ~basic_ifstream, 1251
- adjustfield, 1279
- app, 1279
- ate, 1279
- bad, 1251
- badbit, 1279
- basefield, 1279
- basic_ifstream, 1250, 1251
- beg, 1279
- binary, 1279
- boolalpha, 1280
- clear, 1251
- close, 1252
- copyfmt, 1252
- cur, 1280
- dec, 1280
- end, 1280
- eof, 1252
- eofbit, 1280
- event, 1250
- event_callback, 1248
- exceptions, 1252, 1253
- fail, 1253
- failbit, 1280
- fill, 1253, 1254
- fixed, 1281
- flags, 1254
- floatfield, 1281
- fmtflags, 1248
- gcount, 1255
- get, 1255–1257
- getline, 1257, 1258
- getloc, 1258
- good, 1259
- goodbit, 1281
- hex, 1281
- ignore, 1259
- imbue, 1260
- in, 1281
- init, 1260
- internal, 1281
- iostate, 1249
- is_open, 1261
- isword, 1261
- left, 1282
- narrow, 1261
- oct, 1282
- open, 1261, 1262
- openmode, 1249
- operator bool, 1262
- operator!, 1263
- operator>>, 1263–1265, 1267–1269
- out, 1282
- peek, 1269
- precision, 1269
- putback, 1270
- pword, 1270
- rdbuf, 1271
- rdstate, 1271

- read, 1271
- readsome, 1272
- register_callback, 1273
- right, 1282
- scientific, 1282
- seekdir, 1249
- seekg, 1273
- setf, 1274
- setstate, 1274
- showbase, 1282
- showpoint, 1282
- showpos, 1282
- skipws, 1283
- sync, 1275
- sync_with_stdio, 1275
- tellg, 1276
- tie, 1276
- trunc, 1283
- unget, 1277
- unitbuf, 1283
- unsetf, 1277
- uppercase, 1283
- widen, 1277
- width, 1278
- xalloc, 1278
- std::basic_ios< _CharT, _Traits >, 1284
 - _M_getloc, 1291
 - __ctype_type, 1287
 - __num_get_type, 1287
 - __num_put_type, 1288
 - ~basic_ios, 1290
 - adjustfield, 1302
 - app, 1302
 - ate, 1302
 - bad, 1291
 - badbit, 1302
 - basefield, 1302
 - basic_ios, 1290
 - beg, 1303
 - binary, 1303
 - boolalpha, 1303
 - char_type, 1288
 - clear, 1291
 - copyfmt, 1291
 - cur, 1303
 - dec, 1303
 - end, 1303
 - eof, 1292
 - eofbit, 1303
 - event, 1290
 - event_callback, 1288
 - exceptions, 1292
 - fail, 1293
 - failbit, 1304
 - fill, 1293
 - fixed, 1304
 - flags, 1294
 - floatfield, 1304
 - fmtflags, 1288
 - getloc, 1294
 - good, 1294
 - goodbit, 1304
 - hex, 1304
 - imbue, 1295
 - in, 1304
 - init, 1295
 - int_type, 1289
 - internal, 1305
 - iostate, 1289
 - isword, 1295
 - left, 1305
 - narrow, 1296
 - oct, 1305
 - off_type, 1289
 - openmode, 1289
 - operator bool, 1296
 - operator!, 1296
 - out, 1305
 - pos_type, 1290
 - precision, 1296
 - pword, 1297
 - rdbuf, 1297
 - rdstate, 1298
 - register_callback, 1298
 - right, 1305
 - scientific, 1305
 - seekdir, 1290
 - setf, 1298, 1299
 - setstate, 1299
 - showbase, 1305
 - showpoint, 1305
 - showpos, 1306
 - skipws, 1306
 - sync_with_stdio, 1299
 - tie, 1300
 - traits_type, 1290
 - trunc, 1306
 - unitbuf, 1306
 - unsetf, 1300
 - uppercase, 1306
 - widen, 1301
 - width, 1301
 - xalloc, 1302
- std::basic_istream< _CharT, _Traits >, 1306
 - _M_gcount, 1348
 - _M_getloc, 1315
 - __num_put_type, 1312
 - ~basic_istream, 1314

adjustfield, 1348
app, 1348
ate, 1348
bad, 1315
badbit, 1348
basefield, 1348
basic_istream, 1314
beg, 1348
binary, 1349
boolalpha, 1349
clear, 1315
copyfmt, 1315
cur, 1349
dec, 1349
end, 1349
eof, 1316
eofbit, 1349
event, 1314
event_callback, 1312
exceptions, 1316
fail, 1317
failbit, 1349
fill, 1317
fixed, 1350
flags, 1317, 1318
floatfield, 1350
flush, 1318
fmtflags, 1312
gcount, 1318
get, 1318–1320
getline, 1321, 1322
getloc, 1322
good, 1322
goodbit, 1350
hex, 1350
ignore, 1322, 1323
imbue, 1323
in, 1350
init, 1324
internal, 1351
iostate, 1313
iword, 1324
left, 1351
narrow, 1324
oct, 1351
openmode, 1313
operator bool, 1325
operator!, 1325
operator<<, 1325–1330
operator>>, 1330–1336
out, 1351
peek, 1336
precision, 1336
put, 1337
putback, 1337
pword, 1338
rdbuf, 1338
rdstate, 1339
read, 1339
readsome, 1340
register_callback, 1340
right, 1351
scientific, 1351
seekdir, 1314
seekg, 1340, 1341
seekp, 1341, 1342
setf, 1342, 1343
setstate, 1343
showbase, 1351
showpoint, 1351
showpos, 1351
skipws, 1352
sync, 1343
sync_with_stdio, 1344
tellg, 1344
tellp, 1344
tie, 1345
trunc, 1352
unget, 1345
unitbuf, 1352
unsetf, 1346
uppercase, 1352
widen, 1346
width, 1346
write, 1347
xalloc, 1347
std::basic_istream<_CharT, _Traits >, 1352
 _M_gcount, 1385
 _M_getloc, 1360
 __num_put_type, 1357
 ~basic_istream, 1359
adjustfield, 1385
app, 1385
ate, 1385
bad, 1360
badbit, 1386
basefield, 1386
basic_istream, 1359
beg, 1386
binary, 1386
boolalpha, 1386
clear, 1360
copyfmt, 1360
cur, 1386
dec, 1386
end, 1387
eof, 1361
eofbit, 1387

- event, 1359
- event_callback, 1357
- exceptions, 1361
- fail, 1362
- failbit, 1387
- fill, 1362
- fixed, 1387
- flags, 1362, 1363
- floatfield, 1387
- fmtflags, 1358
- gcount, 1363
- get, 1363–1365
- getline, 1366, 1367
- getloc, 1367
- good, 1367
- goodbit, 1387
- hex, 1388
- ignore, 1367, 1368
- imbue, 1368
- in, 1388
- init, 1369
- internal, 1388
- iostate, 1358
- isword, 1369
- left, 1388
- narrow, 1369
- oct, 1388
- openmode, 1358
- operator bool, 1370
- operator!, 1370
- operator>>, 1370–1375
- out, 1388
- peek, 1376
- precision, 1376
- putback, 1376
- pword, 1377
- rdbuf, 1377, 1378
- rdstate, 1378
- read, 1378
- readsome, 1379
- register_callback, 1379
- right, 1389
- scientific, 1389
- seekdir, 1359
- seekg, 1380
- setf, 1381
- setstate, 1381
- showbase, 1389
- showpoint, 1389
- showpos, 1389
- skipws, 1389
- sync, 1382
- sync_with_stdio, 1382
- tellg, 1382
- tie, 1383
- trunc, 1389
- unget, 1383
- unitbuf, 1389
- unsetf, 1384
- uppercase, 1389
- widen, 1384
- width, 1384
- xalloc, 1385
- std::basic_istream<_CharT, _Traits >::sentry, 2737
- operator bool, 2738
- sentry, 2738
- traits_type, 2738
- std::basic_istream<_CharT, _Traits, _Alloc >, 1390
- _M_gcount, 1426
- _M_getloc, 1398
- __num_put_type, 1395
- ~basic_istream, 1398
- adjustfield, 1427
- app, 1427
- ate, 1427
- bad, 1398
- badbit, 1427
- basefield, 1427
- basic_istream, 1397
- beg, 1427
- binary, 1427
- boolalpha, 1428
- clear, 1398
- copyfmt, 1399
- cur, 1428
- dec, 1428
- end, 1428
- eof, 1399
- eofbit, 1428
- event, 1397
- event_callback, 1395
- exceptions, 1399
- fail, 1401
- failbit, 1428
- fill, 1401
- fixed, 1429
- flags, 1402
- floatfield, 1429
- fmtflags, 1395
- gcount, 1402
- get, 1402–1404
- getline, 1405, 1406
- getloc, 1406
- good, 1406
- goodbit, 1429
- hex, 1429
- ignore, 1406, 1407
- imbue, 1407

- in, [1429](#)
- init, [1408](#)
- internal, [1429](#)
- iostate, [1396](#)
- isword, [1408](#)
- left, [1430](#)
- narrow, [1408](#)
- oct, [1430](#)
- openmode, [1396](#)
- operator bool, [1409](#)
- operator!, [1409](#)
- operator>>, [1409–1414](#)
- out, [1430](#)
- peek, [1415](#)
- precision, [1415](#)
- putback, [1415](#)
- pword, [1416](#)
- rdbuf, [1416](#)
- rdstate, [1418](#)
- read, [1418](#)
- readsome, [1419](#)
- register_callback, [1419](#)
- right, [1430](#)
- scientific, [1430](#)
- seekdir, [1396](#)
- seekg, [1419](#), [1420](#)
- setf, [1420](#), [1422](#)
- setstate, [1422](#)
- showbase, [1430](#)
- showpoint, [1430](#)
- showpos, [1430](#)
- skipws, [1431](#)
- str, [1422](#), [1423](#)
- sync, [1423](#)
- sync_with_stdio, [1423](#)
- tellg, [1424](#)
- tie, [1424](#)
- trunc, [1431](#)
- unget, [1425](#)
- unitbuf, [1431](#)
- unsetf, [1425](#)
- uppercase, [1431](#)
- widen, [1425](#)
- width, [1426](#)
- xalloc, [1426](#)
- std::basic_ofstream<_CharT, _Traits>, [1432](#)
 - _M_getloc, [1440](#)
 - __num_get_type, [1436](#)
 - ~basic_ofstream, [1440](#)
 - adjustfield, [1461](#)
 - app, [1461](#)
 - ate, [1461](#)
 - bad, [1440](#)
 - badbit, [1461](#)
 - basefield, [1461](#)
 - basic_ofstream, [1439](#)
 - beg, [1461](#)
 - binary, [1461](#)
 - boolalpha, [1461](#)
 - clear, [1440](#)
 - close, [1440](#)
 - copyfmt, [1441](#)
 - cur, [1462](#)
 - dec, [1462](#)
 - end, [1462](#)
 - eof, [1441](#)
 - eofbit, [1462](#)
 - event, [1438](#)
 - event_callback, [1436](#)
 - exceptions, [1441](#)
 - fail, [1442](#)
 - failbit, [1462](#)
 - fill, [1442](#)
 - fixed, [1462](#)
 - flags, [1443](#)
 - floatfield, [1463](#)
 - flush, [1443](#)
 - fmtflags, [1437](#)
 - getloc, [1444](#)
 - good, [1444](#)
 - goodbit, [1463](#)
 - hex, [1463](#)
 - imbue, [1444](#)
 - in, [1463](#)
 - init, [1444](#)
 - internal, [1463](#)
 - iostate, [1437](#)
 - is_open, [1445](#)
 - isword, [1445](#)
 - left, [1463](#)
 - narrow, [1445](#)
 - oct, [1464](#)
 - open, [1446](#)
 - openmode, [1438](#)
 - operator bool, [1447](#)
 - operator!, [1447](#)
 - operator<<, [1447–1452](#)
 - out, [1464](#)
 - precision, [1452](#)
 - put, [1454](#)
 - pword, [1454](#)
 - rdbuf, [1454](#), [1455](#)
 - rdstate, [1455](#)
 - register_callback, [1455](#)
 - right, [1464](#)
 - scientific, [1464](#)
 - seekdir, [1438](#)
 - seekp, [1456](#)

- setf, [1456](#), [1457](#)
- setstate, [1457](#)
- showbase, [1464](#)
- showpoint, [1464](#)
- showpos, [1464](#)
- skipws, [1464](#)
- sync_with_stdio, [1457](#)
- tellp, [1458](#)
- tie, [1458](#)
- trunc, [1464](#)
- unitbuf, [1465](#)
- unsetf, [1458](#)
- uppercase, [1465](#)
- widen, [1459](#)
- width, [1459](#)
- write, [1460](#)
- xalloc, [1460](#)
- std::basic_ostream< _CharT, _Traits >, [1465](#)
 - _M_getloc, [1472](#)
 - __num_get_type, [1470](#)
 - ~basic_ostream, [1472](#)
 - adjustfield, [1493](#)
 - app, [1493](#)
 - ate, [1493](#)
 - bad, [1472](#)
 - badbit, [1493](#)
 - basefield, [1493](#)
 - basic_ostream, [1472](#)
 - beg, [1493](#)
 - binary, [1493](#)
 - boolalpha, [1493](#)
 - clear, [1472](#)
 - copyfmt, [1473](#)
 - cur, [1494](#)
 - dec, [1494](#)
 - end, [1494](#)
 - eof, [1473](#)
 - eofbit, [1494](#)
 - event, [1471](#)
 - event_callback, [1470](#)
 - exceptions, [1473](#)
 - fail, [1475](#)
 - failbit, [1494](#)
 - fill, [1475](#)
 - fixed, [1494](#)
 - flags, [1476](#)
 - floatfield, [1495](#)
 - flush, [1476](#)
 - fmtflags, [1470](#)
 - getloc, [1476](#)
 - good, [1477](#)
 - goodbit, [1495](#)
 - hex, [1495](#)
 - imbue, [1477](#)
 - in, [1495](#)
 - init, [1477](#)
 - internal, [1495](#)
 - iostate, [1471](#)
 - isword, [1477](#)
 - left, [1495](#)
 - narrow, [1478](#)
 - oct, [1496](#)
 - openmode, [1471](#)
 - operator bool, [1478](#)
 - operator!, [1478](#)
 - operator<<, [1479](#), [1480](#), [1482–1485](#)
 - out, [1496](#)
 - precision, [1485](#)
 - put, [1485](#)
 - pword, [1486](#)
 - rdbuf, [1486](#), [1487](#)
 - rdstate, [1487](#)
 - register_callback, [1487](#)
 - right, [1496](#)
 - scientific, [1496](#)
 - seekdir, [1471](#)
 - seekp, [1487](#), [1488](#)
 - setf, [1488](#), [1489](#)
 - setstate, [1489](#)
 - showbase, [1496](#)
 - showpoint, [1496](#)
 - showpos, [1496](#)
 - skipws, [1496](#)
 - sync_with_stdio, [1489](#)
 - tellp, [1490](#)
 - tie, [1490](#)
 - trunc, [1496](#)
 - unitbuf, [1497](#)
 - unsetf, [1490](#)
 - uppercase, [1497](#)
 - widen, [1491](#)
 - width, [1491](#)
 - write, [1492](#)
 - xalloc, [1492](#)
- std::basic_ostream< _CharT, _Traits >::sentry, [2739](#)
 - ~sentry, [2739](#)
 - operator bool, [2739](#)
 - sentry, [2739](#)
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1497](#)
 - _M_getloc, [1505](#)
 - __num_get_type, [1502](#)
 - ~basic_ostringstream, [1504](#)
 - adjustfield, [1525](#)
 - app, [1525](#)
 - ate, [1525](#)
 - bad, [1505](#)
 - badbit, [1525](#)
 - basefield, [1525](#)

basic_ostringstream, 1504
beg, 1525
binary, 1525
boolalpha, 1526
clear, 1505
copyfmt, 1505
cur, 1526
dec, 1526
end, 1526
eof, 1506
eofbit, 1526
event, 1504
event_callback, 1502
exceptions, 1506
fail, 1507
failbit, 1526
fill, 1507
fixed, 1527
flags, 1508
floatfield, 1527
flush, 1508
fmtflags, 1502
getloc, 1508
good, 1509
goodbit, 1527
hex, 1527
imbue, 1509
in, 1527
init, 1509
internal, 1527
iostate, 1503
iword, 1509
left, 1528
narrow, 1510
oct, 1528
openmode, 1503
operator bool, 1510
operator!, 1510
operator<<, 1510–1516
out, 1528
precision, 1516
put, 1516
pword, 1517
rdbuf, 1517
rdstate, 1518
register_callback, 1518
right, 1528
scientific, 1528
seekdir, 1503
seekp, 1518, 1519
setf, 1519
setstate, 1521
showbase, 1528
showpoint, 1528
showpos, 1528
skipws, 1529
str, 1521
sync_with_stdio, 1522
tellp, 1522
tie, 1522
trunc, 1529
unitbuf, 1529
unsetf, 1523
uppercase, 1529
widen, 1523
width, 1523, 1524
write, 1524
xalloc, 1524
std::basic_regex<_Ch_type, _Rx_traits >, 1529
 ~basic_regex, 1533
 assign, 1534–1536
 basic_regex, 1531–1533
 flags, 1536
 getloc, 1536
 imbue, 1536
 mark_count, 1537
 operator=, 1537, 1538
 swap, 1538
std::basic_streambuf<_CharT, _Traits >, 1539
 _M_buf_locale, 1555
 _M_in_beg, 1555
 _M_in_cur, 1555
 _M_in_end, 1555
 _M_out_beg, 1555
 _M_out_cur, 1555
 _M_out_end, 1555
 __streambuf_type, 1542
 ~basic_streambuf, 1543
 basic_streambuf, 1543
 char_type, 1542
 eback, 1543
 egptr, 1543
 epptr, 1544
 gbump, 1544
 getloc, 1544
 gptr, 1544
 imbue, 1544
 in_avail, 1545
 int_type, 1542
 off_type, 1542
 overflow, 1545
 pbackfail, 1546
 pbase, 1546
 pbump, 1546
 pos_type, 1542
 pptr, 1547
 pubimbue, 1547
 pubseekoff, 1547

pubseekpos, 1547
 pubsetbuf, 1548
 pubsync, 1548
 sbumpc, 1548
 seekoff, 1548
 seekpos, 1549
 setbuf, 1549
 setg, 1549
 setp, 1550
 sgetc, 1550
 sgetn, 1550
 showmanyc, 1551
 sngetc, 1551
 sputbackc, 1551
 sputc, 1552
 sputn, 1552
 sungetc, 1552
 sync, 1553
 traits_type, 1543
 uflow, 1553
 underflow, 1553
 xsgetc, 1554
 xspn, 1554
 std::basic_string<_CharT, _Traits, _Alloc >, 1590
 ~basic_string, 1600
 append, 1601–1604
 assign, 1604–1608
 at, 1608, 1609
 back, 1609
 basic_string, 1596, 1597, 1599, 1600
 begin, 1610
 c_str, 1610
 capacity, 1610
 cbegin, 1610
 cend, 1610
 clear, 1610
 compare, 1610–1614
 copy, 1615
 crbegin, 1615
 crend, 1615
 data, 1616
 empty, 1616
 end, 1616
 erase, 1616, 1617
 find, 1618, 1619
 find_first_not_of, 1620, 1621
 find_first_of, 1623, 1624
 find_last_not_of, 1626, 1627
 find_last_of, 1629, 1630
 front, 1632
 get_allocator, 1632
 insert, 1632–1637
 length, 1638
 max_size, 1638
 npos, 1656
 operator __sv_type, 1638
 operator+=, 1638, 1639
 operator=, 1640, 1641
 operator[], 1641, 1642
 pop_back, 1642
 push_back, 1642
 rbegin, 1643
 rend, 1643
 replace, 1643–1651
 reserve, 1651
 resize, 1652
 rfind, 1653, 1654
 shrink_to_fit, 1655
 size, 1655
 substr, 1655
 swap, 1656
 std::basic_string_view<_CharT, _Traits >, 1656
 std::basic_stringbuf<_CharT, _Traits, _Alloc >, 1661
 _M_buf_locale, 1677
 _M_in_beg, 1677
 _M_in_cur, 1677
 _M_in_end, 1677
 _M_mode, 1677
 _M_out_beg, 1677
 _M_out_cur, 1677
 _M_out_end, 1678
 basic_stringbuf, 1663
 eback, 1664
 egptr, 1664
 epptr, 1664
 gbump, 1664
 getloc, 1665
 gptr, 1665
 imbue, 1665
 in_avail, 1665
 overflow, 1666
 pbackfail, 1666
 pbase, 1667
 pbump, 1667
 pptr, 1667
 pubimbue, 1667
 pubseekoff, 1668
 pubseekpos, 1668
 pubsetbuf, 1668
 pubsync, 1668
 sbumpc, 1669
 seekoff, 1669
 seekpos, 1669
 setbuf, 1669
 setg, 1670
 setp, 1670
 sgetc, 1671
 sgetn, 1671

- showmanyc, [1671](#)
- snextc, [1672](#)
- sputbackc, [1672](#)
- sputc, [1672](#)
- sputn, [1673](#)
- str, [1673](#)
- sungetc, [1674](#)
- sync, [1674](#)
- uflow, [1674](#)
- underflow, [1674](#)
- xsgetn, [1675](#)
- xspn, [1675](#)
- std::basic_stringstream< _CharT, _Traits, _Alloc >, [1678](#)
 - _M_gcount, [1721](#)
 - _M_getloc, [1687](#)
 - __num_put_type, [1684](#)
 - ~basic_stringstream, [1687](#)
 - adjustfield, [1721](#)
 - app, [1721](#)
 - ate, [1721](#)
 - bad, [1688](#)
 - badbit, [1721](#)
 - basefield, [1721](#)
 - basic_stringstream, [1687](#)
 - beg, [1721](#)
 - binary, [1722](#)
 - boolalpha, [1722](#)
 - clear, [1688](#)
 - copyfmt, [1688](#)
 - cur, [1722](#)
 - dec, [1722](#)
 - end, [1722](#)
 - eof, [1689](#)
 - eofbit, [1722](#)
 - event, [1686](#)
 - event_callback, [1684](#)
 - exceptions, [1689](#)
 - fail, [1690](#)
 - failbit, [1722](#)
 - fill, [1690](#)
 - fixed, [1723](#)
 - flags, [1690](#), [1691](#)
 - floatfield, [1723](#)
 - flush, [1691](#)
 - fmtflags, [1685](#)
 - gcount, [1691](#)
 - get, [1691](#)–[1693](#)
 - getline, [1694](#), [1695](#)
 - getloc, [1695](#)
 - good, [1695](#)
 - goodbit, [1723](#)
 - hex, [1723](#)
 - ignore, [1695](#), [1696](#)
 - imbue, [1696](#)
 - in, [1723](#)
 - init, [1697](#)
 - internal, [1724](#)
 - iostate, [1685](#)
 - isw, [1697](#)
 - left, [1724](#)
 - narrow, [1697](#)
 - oct, [1724](#)
 - openmode, [1686](#)
 - operator bool, [1698](#)
 - operator!, [1698](#)
 - operator<<, [1698](#)–[1703](#)
 - operator>>, [1703](#)–[1709](#)
 - out, [1724](#)
 - peek, [1709](#)
 - precision, [1709](#)
 - put, [1710](#)
 - putback, [1710](#)
 - pwd, [1711](#)
 - rdbuf, [1711](#)
 - rdstate, [1712](#)
 - read, [1712](#)
 - readsome, [1712](#)
 - register_callback, [1713](#)
 - right, [1724](#)
 - scientific, [1724](#)
 - seekdir, [1686](#)
 - seekg, [1713](#), [1714](#)
 - seekp, [1714](#), [1715](#)
 - setf, [1715](#)
 - setstate, [1716](#)
 - showbase, [1724](#)
 - showpoint, [1724](#)
 - showpos, [1724](#)
 - skipws, [1725](#)
 - str, [1716](#)
 - sync, [1716](#)
 - sync_with_stdio, [1717](#)
 - tellg, [1717](#)
 - tellp, [1717](#)
 - tie, [1718](#)
 - trunc, [1725](#)
 - unget, [1718](#)
 - unitbuf, [1725](#)
 - unsetf, [1719](#)
 - uppercase, [1725](#)
 - widen, [1719](#)
 - width, [1719](#)
 - write, [1720](#)
 - xalloc, [1720](#)
- std::bernoulli_distribution, [1725](#)
 - bernoulli_distribution, [1726](#)
 - max, [1726](#)
 - min, [1727](#)

operator(), 1727
 operator==, 1727
 p, 1727
 param, 1727
 reset, 1727
 result_type, 1726
 std::bernoulli_distribution::param_type, 2598
 std::bidirectional_iterator_tag, 1728
 std::binary_function<_Arg1, _Arg2, _Result>, 1740
 first_argument_type, 1741
 result_type, 1741
 second_argument_type, 1741
 std::binary_negate<_Predicate>, 1750
 first_argument_type, 1751
 result_type, 1751
 second_argument_type, 1751
 std::binder1st<_Operation>, 1752
 argument_type, 1752
 result_type, 1752
 std::binder2nd<_Operation>, 1753
 argument_type, 1754
 result_type, 1754
 std::binomial_distribution<_IntType>, 1754
 max, 1755
 min, 1755
 operator<<, 1756
 operator>>, 1757
 operator(), 1755
 operator==, 1757
 p, 1756
 param, 1756
 reset, 1756
 result_type, 1755
 t, 1756
 std::binomial_distribution<_IntType>::param_type, 2599
 std::bitset<_Nb>, 1765
 all, 1769
 any, 1770
 bitset, 1768, 1769
 count, 1770
 flip, 1770
 none, 1770
 operator<<, 1771
 operator<=, 1771
 operator>>, 1771
 operator>=, 1771
 operator~, 1773
 operator^=, 1772
 operator==, 1771
 operator&=, 1770
 operator[], 1772
 operator|=, 1773
 reset, 1773
 set, 1773, 1774
 size, 1774
 test, 1774
 to_string, 1774
 to_ulong, 1775
 std::bitset<_Nb>::reference, 2687
 std::cauchy_distribution<_RealType>, 1778
 max, 1779
 min, 1779
 operator(), 1779
 operator==, 1780
 param, 1780
 reset, 1780
 result_type, 1779
 std::cauchy_distribution<_RealType>::param_type, 2599
 std::char_traits<__gnu_cxx::character<_Value, _Int, _St>>, 1796
 std::char_traits<_CharT>, 1795
 std::char_traits<char>, 1797
 std::char_traits<wchar_t>, 1798
 std::chi_squared_distribution<_RealType>, 1799
 max, 1800
 min, 1800
 operator<<, 1801
 operator>>, 1801
 operator(), 1800
 operator==, 1801
 param, 1800
 reset, 1801
 result_type, 1800
 std::chi_squared_distribution<_RealType>::param_type, 2600
 std::chrono, 741
 std::chrono::_V2::steady_clock, 2840
 std::chrono::_V2::system_clock, 2866
 std::chrono::duration<_Rep, _Period>, 2002
 std::chrono::duration_values<_Rep>, 2003
 std::chrono::time_point<_Clock, _Dur>, 2903
 std::chrono::treat_as_floating_point<_Rep>, 2912
 std::codecvt<_InternT, _ExternT, _StateT>, 1802
 do_always_noconv, 1804
 do_encoding, 1804
 do_in, 1804
 do_length, 1804
 do_max_length, 1804
 do_out, 1804
 do_unshift, 1805
 in, 1805
 out, 1806
 unshift, 1806
 std::codecvt<_InternT, _ExternT, encoding_state>, 1807
 do_always_noconv, 1809
 do_encoding, 1809
 do_in, 1809

- do_length, 1809
- do_max_length, 1809
- do_out, 1809
- do_unshift, 1810
- in, 1810
- out, 1811
- unshift, 1811
- std::codecvt< char, char, mbstate_t >, 1812
 - do_always_noconv, 1814
 - do_encoding, 1814
 - do_in, 1814
 - do_length, 1814
 - do_max_length, 1814
 - do_out, 1814
 - do_unshift, 1815
 - in, 1815
 - out, 1816
 - unshift, 1816
- std::codecvt< char16_t, char, mbstate_t >, 1817
 - do_always_noconv, 1818
 - do_encoding, 1818
 - do_in, 1819
 - do_length, 1819
 - do_max_length, 1819
 - do_out, 1819
 - do_unshift, 1819
 - in, 1820
 - out, 1820
 - unshift, 1821
- std::codecvt< char32_t, char, mbstate_t >, 1822
 - do_always_noconv, 1823
 - do_encoding, 1823
 - do_in, 1823
 - do_length, 1823
 - do_max_length, 1824
 - do_out, 1824
 - do_unshift, 1824
 - in, 1824
 - out, 1825
 - unshift, 1826
- std::codecvt< wchar_t, char, mbstate_t >, 1827
 - do_always_noconv, 1828
 - do_encoding, 1828
 - do_in, 1828
 - do_length, 1829
 - do_max_length, 1829
 - do_out, 1829
 - do_unshift, 1829
 - in, 1829
 - out, 1830
 - unshift, 1831
- std::codecvt_base, 1832
- std::codecvt_byname< _InternT, _ExternT, _StateT >, 1832
 - do_always_noconv, 1834
 - do_encoding, 1834
 - do_in, 1834
 - do_length, 1835
 - do_max_length, 1835
 - do_out, 1835
 - do_unshift, 1835
 - in, 1835
 - out, 1836
 - unshift, 1837
- std::collate< _CharT >, 1838
 - ~collate, 1840
 - char_type, 1839
 - collate, 1839, 1840
 - compare, 1840
 - do_compare, 1840
 - do_hash, 1841
 - do_transform, 1841
 - hash, 1843
 - id, 1844
 - string_type, 1839
 - transform, 1843
- std::collate_byname< _CharT >, 1844
 - char_type, 1845
 - compare, 1846
 - do_compare, 1846
 - do_hash, 1846
 - do_transform, 1847
 - hash, 1847
 - id, 1848
 - string_type, 1845
 - transform, 1848
- std::common_iterator< _It, _Sent >, 1848
- std::common_reference_with, 791
- std::common_type< _Tp >, 1850
- std::common_type< chrono::duration< _Rep, _Period > >, 1850
- std::common_type< chrono::duration< _Rep, _Period >, chrono::duration< _Rep, _Period > >, 1850
- std::common_type< chrono::duration< _Rep1, _Period1 >, chrono::duration< _Rep2, _Period2 > >, 1850
- std::common_type< chrono::time_point< _Clock, _Duration > >, 1851
- std::common_type< chrono::time_point< _Clock, _Duration >, chrono::time_point< _Clock, _Duration > >, 1851
- std::common_type< chrono::time_point< _Clock, _Duration1 >, chrono::time_point< _Clock, _Duration2 > >, 1851
- std::common_with, 791
- std::compare_three_way_result< _Tp, _Up >, 1852
- std::complex< _Tp >, 1852
 - complex, 1853

- operator+=, 1853
- operator-=, 1853
- value_type, 1853
- std::complex< double >, 1854
- std::complex< float >, 1854
- std::complex< long double >, 1855
- std::condition_variable, 1858
- std::conditional< _Cond, _Iftrue, _Iffalse >, 1859
- std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >, 1861
 - first_argument_type, 1861
 - result_type, 1861
 - second_argument_type, 1861
- std::const_mem_fun1_t< _Ret, _Tp, _Arg >, 1862
 - first_argument_type, 1862
 - result_type, 1862
 - second_argument_type, 1862
- std::const_mem_fun_ref_t< _Ret, _Tp >, 1863
 - argument_type, 1863
 - result_type, 1863
- std::const_mem_fun_t< _Ret, _Tp >, 1864
 - argument_type, 1864
 - result_type, 1864
- std::constructible_from, 791
- std::contiguous_iterator_tag, 1882
- std::convertible_to, 792
- std::copy_constructible, 792
- std::counted_iterator< _It >, 1883
- std::ctype< _CharT >, 1885
 - do_is, 1887
 - do_narrow, 1887, 1888
 - do_scan_is, 1888
 - do_scan_not, 1889
 - do_tolower, 1889, 1890
 - do_toupper, 1890, 1891
 - do_widen, 1891, 1892
 - id, 1898
 - is, 1892, 1893
 - narrow, 1893, 1894
 - scan_is, 1894
 - scan_not, 1895
 - tolower, 1895
 - toupper, 1896
 - widen, 1897
- std::ctype< char >, 1898
 - ~ctype, 1901
 - char_type, 1900
 - classic_table, 1901
 - ctype, 1900
 - do_narrow, 1901
 - do_tolower, 1902
 - do_toupper, 1903
 - do_widen, 1903, 1904
 - id, 1909
 - is, 1904, 1905
 - narrow, 1905
 - scan_is, 1906
 - scan_not, 1906
 - table, 1907
 - table_size, 1909
 - tolower, 1907
 - toupper, 1908
 - widen, 1908, 1909
- std::ctype< wchar_t >, 1910
 - ~ctype, 1912
 - char_type, 1912
 - ctype, 1912
 - do_is, 1912, 1913
 - do_narrow, 1913, 1914
 - do_scan_is, 1914
 - do_scan_not, 1915
 - do_tolower, 1915, 1916
 - do_toupper, 1916
 - do_widen, 1917
 - id, 1923
 - is, 1918
 - narrow, 1919
 - scan_is, 1919
 - scan_not, 1920
 - tolower, 1920, 1921
 - toupper, 1921
 - widen, 1922
- std::ctype_base, 1923
- std::ctype_byname< _CharT >, 1924
 - do_is, 1926
 - do_narrow, 1926, 1927
 - do_scan_is, 1927
 - do_scan_not, 1928
 - do_tolower, 1928, 1929
 - do_toupper, 1929, 1930
 - do_widen, 1930, 1931
 - id, 1937
 - is, 1931, 1932
 - narrow, 1932, 1933
 - scan_is, 1933
 - scan_not, 1934
 - tolower, 1934
 - toupper, 1935
 - widen, 1936
- std::ctype_byname< char >, 1937
 - char_type, 1939
 - classic_table, 1939
 - do_narrow, 1939
 - do_tolower, 1940
 - do_toupper, 1941
 - do_widen, 1941, 1942
 - id, 1948
 - is, 1942, 1943
 - narrow, 1943

- scan_is, 1944
- scan_not, 1944
- table, 1945
- table_size, 1948
- tolower, 1945
- toupper, 1946
- widen, 1947
- std::decay< _Tp >, 1949
- std::decimal, 745
 - decimal32_to_long_long, 754
- std::decimal::decimal128, 1949
 - decimal128, 1951
- std::decimal::decimal32, 1951
 - decimal32, 1952
- std::decimal::decimal64, 1952
 - decimal64, 1954
- std::default_delete< _Tp >, 1955
 - default_delete, 1955
 - operator(), 1955
- std::default_delete< _Tp[] >, 1955
 - default_delete, 1956
 - operator(), 1956
- std::default_initializable, 792
- std::default_sentinel_t, 1959
- std::defer_lock_t, 1961
- std::deque< _Tp, _Alloc >, 1963
 - _M_fill_initialize, 1971
 - _M_new_elements_at_back, 1971
 - _M_new_elements_at_front, 1971
 - _M_pop_back_aux, 1972
 - _M_pop_front_aux, 1972
 - _M_push_back_aux, 1972
 - _M_push_front_aux, 1972
 - _M_range_check, 1972
 - _M_range_initialize, 1972, 1973
 - _M_reallocate_map, 1973
 - _M_reserve_elements_at_back, 1973
 - _M_reserve_elements_at_front, 1974
 - _M_reserve_map_at_back, 1974
 - _M_reserve_map_at_front, 1974
 - ~deque, 1971
 - assign, 1974, 1975
 - at, 1975, 1976
 - back, 1976
 - begin, 1976
 - cbegin, 1977
 - cend, 1977
 - clear, 1977
 - crbegin, 1977
 - crend, 1977
 - deque, 1968–1970
 - emplace, 1977
 - empty, 1978
 - end, 1978
 - erase, 1978
 - front, 1980
 - get_allocator, 1980
 - insert, 1980–1982
 - max_size, 1982
 - operator=, 1982, 1983
 - operator[], 1983, 1984
 - pop_back, 1984
 - pop_front, 1984
 - push_back, 1984
 - push_front, 1985
 - rbegin, 1985
 - rend, 1985
 - resize, 1985, 1986
 - shrink_to_fit, 1986
 - size, 1986
 - swap, 1986
- std::derived_from, 792
- std::destroying_delete_t, 1987
- std::destructible, 792
- std::discard_block_engine< _RandomNumberEngine, __p, __r >, 1991
 - base, 1994
 - discard, 1994
 - discard_block_engine, 1993
 - max, 1994
 - min, 1994
 - operator<<, 1995
 - operator>>, 1996
 - operator(), 1994
 - operator==, 1995
 - result_type, 1992
 - seed, 1994, 1995
- std::discrete_distribution< _IntType >, 1996
 - max, 1997
 - min, 1997
 - operator<<, 1998
 - operator>>, 1999
 - operator(), 1998
 - operator==, 1999
 - param, 1998
 - probabilities, 1998
 - reset, 1998
 - result_type, 1997
- std::discrete_distribution< _IntType >::param_type, 2600
- std::divides< _Tp >, 1999
 - first_argument_type, 2000
 - result_type, 2000
 - second_argument_type, 2000
- std::divides< void >, 2001
- std::domain_error, 2001
 - what, 2002
- std::enable_if< bool, _Tp >, 2019
- std::enable_shared_from_this< _Tp >, 2019

weak_from_this, 2020
 std::equal_to< _Tp >, 2044
 first_argument_type, 2045
 result_type, 2045
 second_argument_type, 2045
 std::equal_to< void >, 2046
 std::equivalence_relation, 793
 std::error_code, 2048
 category, 2048
 default_error_condition, 2048
 message, 2049
 operator bool, 2049
 value, 2049
 std::error_condition, 2049
 assign, 2050
 category, 2050
 clear, 2050
 error_condition, 2050
 message, 2050
 operator bool, 2050
 value, 2050
 std::exception, 2052
 what, 2053
 std::experimental, 754
 gcd, 764
 get_deleter, 764
 is_bind_expression_v, 766
 is_placeholder_v, 766
 lcm, 764
 make_boyer_moore_horspool_searcher, 764
 make_boyer_moore_searcher, 765
 make_default_searcher, 765
 make_ostream_joiner, 765
 not_fn, 765
 sample, 765
 std::experimental::filesystem::v1::filesystem_error, 2064
 what, 2065
 std::experimental::filesystem::v1::path, 2614
 std::experimental::filesystem::v1::path::iterator, 2257
 std::experimental::filesystem::v1::space_info, 2796
 std::experimental::fundamentals_v1::any, 1115
 ~any, 1116
 any, 1115, 1116
 clear, 1116
 empty, 1116
 operator=, 1116, 1117
 swap, 1117
 type, 1117
 std::experimental::fundamentals_v1::bad_any_cast, 1160
 what, 1160
 std::experimental::fundamentals_v1::bad_optional_access, 1164
 what, 1165
 std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >, 1658
 std::experimental::fundamentals_v1::in_place_t, 2171
 std::experimental::fundamentals_v1::nullopt_t, 2499
 std::experimental::fundamentals_v1::optional< _Tp >, 2564
 std::experimental::fundamentals_v2::ostream_joiner< _DelimT, _CharT, _Traits >, 2571
 std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >, 2585
 first_argument_type, 2585
 result_type, 2585
 second_argument_type, 2585
 std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >, 2587
 first_argument_type, 2588
 result_type, 2588
 second_argument_type, 2588
 std::experimental::fundamentals_v2::propagate_const< _Tp >, 2648
 std::exponential_distribution< _RealType >, 2054
 exponential_distribution, 2055
 lambda, 2055
 max, 2055
 min, 2055
 operator(), 2055
 operator==, 2056
 param, 2055, 2056
 reset, 2056
 result_type, 2055
 std::exponential_distribution< _RealType >::param_type, 2601
 std::extent< typename, _UInt >, 2057
 std::extreme_value_distribution< _RealType >, 2057
 a, 2058
 b, 2058
 max, 2059
 min, 2059
 operator(), 2059
 operator==, 2060
 param, 2059
 reset, 2059
 result_type, 2058
 std::extreme_value_distribution< _RealType >::param_type, 2601
 std::filesystem, 766
 std::filesystem::__directory_iterator_proxy, 837
 std::filesystem::directory_entry, 1989
 std::filesystem::directory_iterator, 1990
 std::filesystem::file_status, 2063
 std::filesystem::filesystem_error, 2065
 what, 2066
 std::filesystem::path, 2616
 format, 2619

- operator!=, 2619
- operator<, 2619
- operator<<, 2619
- operator<=, 2619
- operator>, 2619
- operator>>, 2620
- operator>=, 2620
- operator/, 2619
- operator==, 2619
- std::filesystem::path::iterator, 2257
- std::filesystem::recursive_directory_iterator, 2685
- std::filesystem::space_info, 2796
- std::fisher_f_distribution< _RealType >, 2066
 - max, 2068
 - min, 2068
 - operator<<, 2069
 - operator>>, 2069
 - operator(), 2068
 - operator==, 2069
 - param, 2068
 - reset, 2068
 - result_type, 2068
- std::fisher_f_distribution< _RealType >::param_type, 2602
- std::forward_iterator_tag, 2070
- std::forward_list< _Tp, _Alloc >, 2074
 - ~forward_list, 2079
 - assign, 2079, 2080
 - before_begin, 2080
 - begin, 2080
 - cbegin, 2081
 - cend, 2081
 - clear, 2081
 - emplace_after, 2081
 - emplace_front, 2081
 - empty, 2082
 - end, 2082
 - erase_after, 2082, 2083
 - forward_list, 2076–2078
 - front, 2083
 - get_allocator, 2083
 - insert_after, 2083–2085
 - max_size, 2085
 - merge, 2085
 - operator=, 2086
 - pop_front, 2087
 - push_front, 2087
 - remove, 2087
 - remove_if, 2087
 - resize, 2088
 - reverse, 2088
 - sort, 2088, 2089
 - splice_after, 2089, 2090
 - swap, 2090
 - unique, 2090, 2091
- std::fpos< _StateT >, 2091
 - fpos, 2092
 - operator streamoff, 2092
 - operator+, 2092
 - operator+=, 2092
 - operator-, 2092
 - operator-=, 2092
 - state, 2093
- std::from_chars_result, 2094
- std::front_insert_iterator< _Container >, 2095
 - container_type, 2096
 - front_insert_iterator, 2096
 - iterator_category, 2096
 - operator*, 2096
 - operator++, 2096
 - operator=, 2097
 - pointer, 2096
 - reference, 2096
 - value_type, 2096
- std::function< _Res(_ArgTypes...)>, 2097
 - function, 2098, 2099
 - operator bool, 2100
 - operator(), 2100
 - operator=, 2100–2102
 - swap, 2102
 - target, 2102
 - target_type, 2103
- std::future< _Res >, 2103
 - _M_get_result, 2104
 - future, 2104
 - get, 2104
- std::future< _Res & >, 2105
 - _M_get_result, 2106
 - future, 2106
 - get, 2106
- std::future< void >, 2106
 - _M_get_result, 2107
 - future, 2107
 - get, 2107
- std::future_error, 2108
 - what, 2108
- std::gamma_distribution< _RealType >, 2108
 - alpha, 2110
 - beta, 2110
 - gamma_distribution, 2110
 - max, 2110
 - min, 2110
 - operator<<, 2111
 - operator>>, 2112
 - operator(), 2110, 2111
 - operator==, 2112
 - param, 2111

- reset, 2111
- result_type, 2110
- std::gamma_distribution< _RealType >::param_type, 2602
- std::geometric_distribution< _IntType >, 2112
 - max, 2113
 - min, 2113
 - operator(), 2114
 - operator==, 2114
 - p, 2114
 - param, 2114
 - reset, 2114
 - result_type, 2113
- std::geometric_distribution< _IntType >::param_type, 2603
- std::greater< _Tp >, 2125
 - first_argument_type, 2126
 - result_type, 2126
 - second_argument_type, 2126
- std::greater< void >, 2126
- std::greater_equal< _Tp >, 2127
 - first_argument_type, 2128
 - result_type, 2128
 - second_argument_type, 2128
- std::greater_equal< void >, 2128
- std::gslice, 2129
- std::gslice_array< _Tp >, 2130
 - operator<=, 2132
 - operator>=, 2132
 - operator*=, 2132
 - operator^=, 2132
 - operator+=, 2132
 - operator-=, 2132
 - operator/=: 2132
 - operator%=: 2131
 - operator&=: 2131
 - operator|=, 2132
- std::has_unique_object_representations< _Tp >, 2133
- std::has_virtual_destructor< _Tp >, 2134
- std::hash< __debug::bitset< _Nb > >, 2135
- std::hash< __debug::vector< bool, _Alloc > >, 2135
- std::hash< __gnu_cxx::__u16vstring >, 2135
- std::hash< __gnu_cxx::__u32vstring >, 2136
- std::hash< __gnu_cxx::__vstring >, 2136
- std::hash< __gnu_cxx::__wvstring >, 2137
- std::hash< __gnu_cxx::throw_value_limit >, 2137
 - argument_type, 2138
 - result_type, 2138
- std::hash< __gnu_cxx::throw_value_random >, 2138
 - argument_type, 2138
 - result_type, 2139
- std::hash< __gnu_debug::basic_string< _CharT > >, 2139
- std::hash< __shared_ptr< _Tp, _Lp > >, 2139
- std::hash< _Tp >, 2134
- std::hash< _Tp * >, 2140
- std::hash< bool >, 2140
- std::hash< char >, 2141
- std::hash< char16_t >, 2141
- std::hash< char32_t >, 2141
- std::hash< double >, 2142
- std::hash< error_code >, 2142
- std::hash< error_condition >, 2142
- std::hash< experimental::optional< _Tp > >, 2143
- std::hash< experimental::shared_ptr< _Tp > >, 2143
- std::hash< float >, 2144
- std::hash< int >, 2144
- std::hash< long >, 2144
- std::hash< long double >, 2145
- std::hash< long long >, 2145
- std::hash< shared_ptr< _Tp > >, 2145
- std::hash< short >, 2146
- std::hash< signed char >, 2146
- std::hash< string >, 2147
- std::hash< thread::id >, 2147
- std::hash< type_index >, 2147
- std::hash< u16string >, 2148
- std::hash< u32string >, 2148
- std::hash< unique_ptr< _Tp, _Dp > >, 2148
- std::hash< unsigned char >, 2149
- std::hash< unsigned int >, 2149
- std::hash< unsigned long >, 2150
- std::hash< unsigned long long >, 2150
- std::hash< unsigned short >, 2150
- std::hash< wchar_t >, 2151
- std::hash< wstring >, 2151
- std::hash<::bitset< _Nb > >, 2151
- std::hash<::vector< bool, _Alloc > >, 2152
- std::identity, 2171
- std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2172
 - base, 2174
 - discard, 2174
 - independent_bits_engine, 2173
 - max, 2174
 - min, 2174
 - operator>=, 2175
 - operator(), 2174
 - operator==, 2175
 - result_type, 2172
 - seed, 2174, 2175
- std::indirect_array< _Tp >, 2176
 - operator<=, 2178
 - operator>=, 2178
 - operator*=, 2177
 - operator^=, 2178
 - operator+=, 2177
 - operator-=, 2178

- operator/=[, 2178](#)
- operator%=[, 2177](#)
- operator&=[, 2177](#)
- operator|=[, 2178](#)
- std::indirectly_comparable[, 793](#)
- std::indirectly_copyable[, 793](#)
- std::indirectly_movable[, 793](#)
- std::indirectly_readable[, 793](#)
- std::indirectly_swappable[, 794](#)
- std::indirectly_writable[, 794](#)
- std::initializer_list<_E>[, 2178](#)
 - begin[, 2179](#)
 - end[, 2179](#)
- std::input_iterator_tag[, 2180](#)
- std::insert_iterator<_Container>[, 2181](#)
 - container_type[, 2183](#)
 - insert_iterator[, 2183](#)
 - iterator_category[, 2183](#)
 - operator*[, 2183](#)
 - operator++[, 2183](#)
 - operator=[, 2184](#)
 - pointer[, 2183](#)
 - reference[, 2183](#)
 - value_type[, 2183](#)
- std::integer_sequence<_Tp, _Idx>[, 2184](#)
- std::integral_constant<_Tp, __v>[, 2185](#)
- std::invalid_argument[, 2186](#)
 - what[, 2187](#)
- std::invocable[, 794](#)
- std::invoke_result<_Functor, _ArgTypes>[, 2187](#)
- std::ios_base[, 2187](#)
 - _M_getloc[, 2192](#)
 - ~ios_base[, 2192](#)
 - adjustfield[, 2197](#)
 - app[, 2197](#)
 - ate[, 2198](#)
 - badbit[, 2198](#)
 - basefield[, 2198](#)
 - beg[, 2198](#)
 - binary[, 2198](#)
 - boolalpha[, 2198](#)
 - cur[, 2198](#)
 - dec[, 2199](#)
 - end[, 2199](#)
 - eofbit[, 2199](#)
 - event[, 2191](#)
 - event_callback[, 2190](#)
 - failbit[, 2199](#)
 - fixed[, 2199](#)
 - flags[, 2192](#)
 - floatfield[, 2199](#)
 - fmtflags[, 2190](#)
 - getloc[, 2192](#)
 - goodbit[, 2200](#)
 - hex[, 2200](#)
 - imbue[, 2193](#)
 - in[, 2200](#)
 - internal[, 2200](#)
 - iostate[, 2190](#)
 - iword[, 2193](#)
 - left[, 2200](#)
 - oct[, 2200](#)
 - openmode[, 2191](#)
 - out[, 2201](#)
 - precision[, 2193](#)
 - pword[, 2194](#)
 - register_callback[, 2194](#)
 - right[, 2201](#)
 - scientific[, 2201](#)
 - seekdir[, 2191](#)
 - setf[, 2194, 2196](#)
 - showbase[, 2201](#)
 - showpoint[, 2201](#)
 - showpos[, 2201](#)
 - skipws[, 2201](#)
 - sync_with_stdio[, 2196](#)
 - trunc[, 2201](#)
 - unitbuf[, 2201](#)
 - unsetf[, 2196](#)
 - uppercase[, 2202](#)
 - width[, 2197](#)
 - xalloc[, 2197](#)
- std::ios_base::failure[, 2062](#)
 - what[, 2063](#)
- std::is_abstract<_Tp>[, 2202](#)
- std::is_aggregate<_Tp>[, 2203](#)
- std::is_arithmetic<_Tp>[, 2203](#)
- std::is_array<typename>[, 2204](#)
- std::is_assignable<_Tp, _Up>[, 2204](#)
- std::is_base_of<_Base, _Derived>[, 2205](#)
- std::is_bind_expression<_Bind<_Signature>>[, 2207](#)
- std::is_bind_expression<_Bind_result<_Result, _Signature>>[, 2208](#)
- std::is_bind_expression<_Tp>[, 2206](#)
- std::is_bind_expression<const _Bind<_Signature>>[, 2209](#)
- std::is_bind_expression<const _Bind_result<_Result, _Signature>>[, 2210](#)
- std::is_bind_expression<const volatile _Bind<_Signature>>[, 2211](#)
- std::is_bind_expression<const volatile _Bind_result<_Result, _Signature>>[, 2212](#)
- std::is_bind_expression<volatile _Bind<_Signature>>[, 2213](#)
- std::is_bind_expression<volatile _Bind_result<_Result, _Signature>>[, 2214](#)
- std::is_bounded_array<_Tp>[, 2214](#)
- std::is_class<_Tp>[, 2215](#)

- `std::is_compound< _Tp >`, 2215
- `std::is_const< typename >`, 2216
- `std::is_constructible< _Tp, _Args >`, 2216
- `std::is_convertible< _From, _To >`, 2217
- `std::is_copy_assignable< _Tp >`, 2217
- `std::is_copy_constructible< _Tp >`, 2217
- `std::is_default_constructible< _Tp >`, 2217
- `std::is_destructible< _Tp >`, 2218
- `std::is_empty< _Tp >`, 2218
- `std::is_enum< _Tp >`, 2219
- `std::is_error_code_enum< _Tp >`, 2220
- `std::is_error_code_enum< future_errc >`, 2221
- `std::is_error_condition_enum< _Tp >`, 2222
- `std::is_final< _Tp >`, 2223
- `std::is_floating_point< _Tp >`, 2223
- `std::is_function< _Tp >`, 2224
- `std::is_fundamental< _Tp >`, 2224
- `std::is_integral< _Tp >`, 2224
- `std::is_invocable< _Fn, _ArgTypes >`, 2224
- `std::is_invocable_r< _Ret, _Fn, _ArgTypes >`, 2225
- `std::is_layout_compatible< _Tp, _Up >`, 2225
- `std::is_literal_type< _Tp >`, 2226
- `std::is_lvalue_reference< typename >`, 2227
- `std::is_member_function_pointer< _Tp >`, 2228
- `std::is_member_object_pointer< _Tp >`, 2229
- `std::is_member_pointer< _Tp >`, 2229
- `std::is_move_assignable< _Tp >`, 2230
- `std::is_move_constructible< _Tp >`, 2230
- `std::is_nothrow_assignable< _Tp, _Up >`, 2230
- `std::is_nothrow_constructible< _Tp, _Args >`, 2230
- `std::is_nothrow_convertible< _From, _To >`, 2231
- `std::is_nothrow_copy_assignable< _Tp >`, 2231
- `std::is_nothrow_copy_constructible< _Tp >`, 2231
- `std::is_nothrow_default_constructible< _Tp >`, 2231
- `std::is_nothrow_destructible< _Tp >`, 2231
- `std::is_nothrow_invocable< _Fn, _ArgTypes >`, 2232
- `std::is_nothrow_invocable_r< _Ret, _Fn, _ArgTypes >`, 2232
- `std::is_nothrow_move_assignable< _Tp >`, 2232
- `std::is_nothrow_move_constructible< _Tp >`, 2232
- `std::is_nothrow_swappable< _Tp >`, 2233
- `std::is_nothrow_swappable_with< _Tp, _Up >`, 2233
- `std::is_null_pointer< _Tp >`, 2234
- `std::is_object< _Tp >`, 2234
- `std::is_placeholder< _Placeholder< _Num > >`, 2236
- `std::is_placeholder< _Tp >`, 2235
- `std::is_pod< _Tp >`, 2237
- `std::is_pointer< _Tp >`, 2238
- `std::is_pointer_interconvertible_base_of< _Base, _Derived >`, 2239
- `std::is_polymorphic< _Tp >`, 2240
- `std::is_reference< _Tp >`, 2240
- `std::is_rvalue_reference< typename >`, 2241
- `std::is_same< _Tp, _Up >`, 2242
- `std::is_scalar< _Tp >`, 2242
- `std::is_signed< _Tp >`, 2243
- `std::is_standard_layout< _Tp >`, 2243
- `std::is_swappable< _Tp >`, 2244
- `std::is_swappable_with< _Tp, _Up >`, 2244
- `std::is_trivial< _Tp >`, 2244
- `std::is_trivially_assignable< _Tp, _Up >`, 2245
- `std::is_trivially_constructible< _Tp, _Args >`, 2245
- `std::is_trivially_copy_assignable< _Tp >`, 2245
- `std::is_trivially_copy_constructible< _Tp >`, 2245
- `std::is_trivially_copyable< _Tp >`, 2246
- `std::is_trivially_default_constructible< _Tp >`, 2247
- `std::is_trivially_destructible< _Tp >`, 2247
- `std::is_trivially_move_assignable< _Tp >`, 2247
- `std::is_trivially_move_constructible< _Tp >`, 2247
- `std::is_unbounded_array< _Tp >`, 2247
- `std::is_union< _Tp >`, 2248
- `std::is_unsigned< _Tp >`, 2249
- `std::is_void< _Tp >`, 2249
- `std::is_volatile< typename >`, 2249
- `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >`, 2250
 - difference_type, 2251
 - istream_iterator, 2252
 - iterator_category, 2251
 - operator==, 2252
 - pointer, 2251
 - reference, 2251
 - value_type, 2251
- `std::istreambuf_iterator< _CharT, _Traits >`, 2252
 - char_type, 2254
 - difference_type, 2254
 - equal, 2255
 - int_type, 2254
 - istream_type, 2254
 - istreambuf_iterator, 2255
 - iterator_category, 2254
 - operator*, 2255
 - operator++, 2255
 - pointer, 2254
 - reference, 2254
 - streambuf_type, 2254
 - traits_type, 2254
 - value_type, 2254
- `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`, 2258
 - difference_type, 2259
 - iterator_category, 2259
 - pointer, 2260
 - reference, 2260
 - value_type, 2260
- `std::iterator_traits< _Iterator >`, 2260
- `std::iterator_traits< _Tp * >`, 2260
- `std::jthread`, 2262
- `std::length_error`, 2271

- what, [2272](#)
- std::less< _Tp >, [2272](#)
 - first_argument_type, [2272](#)
 - result_type, [2272](#)
 - second_argument_type, [2273](#)
- std::less< void >, [2273](#)
- std::less_equal< _Tp >, [2274](#)
 - first_argument_type, [2274](#)
 - result_type, [2274](#)
 - second_argument_type, [2274](#)
- std::less_equal< void >, [2275](#)
- std::linear_congruential_engine< _UIntType, __a, __c, __m >, [2276](#)
 - discard, [2278](#)
 - increment, [2281](#)
 - linear_congruential_engine, [2278](#)
 - max, [2278](#)
 - min, [2278](#)
 - modulus, [2281](#)
 - multiplier, [2281](#)
 - operator<<, [2280](#)
 - operator>>, [2280](#)
 - operator(), [2279](#)
 - operator==, [2280](#)
 - result_type, [2277](#)
 - seed, [2279](#)
- std::list< _Tp, _Alloc >, [2284](#)
 - _M_create_node, [2291](#)
 - ~list, [2290](#)
 - assign, [2291](#), [2292](#)
 - back, [2292](#)
 - begin, [2292](#)
 - cbegin, [2292](#)
 - cend, [2293](#)
 - clear, [2293](#)
 - crbegin, [2293](#)
 - crend, [2293](#)
 - emplace, [2293](#)
 - empty, [2294](#)
 - end, [2294](#)
 - erase, [2294](#)
 - front, [2295](#)
 - get_allocator, [2295](#)
 - insert, [2295](#)–[2297](#)
 - list, [2288](#)–[2290](#)
 - max_size, [2297](#)
 - merge, [2297](#), [2299](#)
 - operator=, [2299](#), [2300](#)
 - pop_back, [2300](#)
 - pop_front, [2300](#)
 - push_back, [2300](#)
 - push_front, [2301](#)
 - rbegin, [2301](#)
 - remove, [2301](#)
 - remove_if, [2302](#)
 - rend, [2302](#)
 - resize, [2302](#), [2303](#)
 - reverse, [2303](#)
 - size, [2303](#)
 - sort, [2303](#)
 - splice, [2303](#)–[2305](#)
 - swap, [2305](#)
 - unique, [2306](#)
- std::literals, [769](#)
- std::literals::chrono_literals, [770](#)
- std::locale, [2308](#)
 - ~locale, [2312](#)
 - all, [2315](#)
 - category, [2309](#)
 - classic, [2312](#)
 - collate, [2315](#)
 - combine, [2312](#)
 - ctype, [2316](#)
 - global, [2313](#)
 - has_facet, [2314](#)
 - locale, [2310](#)–[2312](#)
 - messages, [2316](#)
 - monetary, [2316](#)
 - name, [2313](#)
 - none, [2316](#)
 - numeric, [2316](#)
 - operator(), [2313](#)
 - operator=, [2314](#)
 - operator==, [2314](#)
 - time, [2316](#)
 - use_facet, [2315](#)
- std::locale::facet, [2060](#)
 - ~facet, [2062](#)
 - facet, [2062](#)
- std::locale::id, [2169](#)
 - has_facet, [2169](#)
 - id, [2169](#)
 - use_facet, [2170](#)
- std::lock_guard< _Mutex >, [2317](#)
- std::logic_error, [2317](#)
 - logic_error, [2318](#)
 - what, [2318](#)
- std::logical_and< _Tp >, [2318](#)
 - first_argument_type, [2319](#)
 - result_type, [2319](#)
 - second_argument_type, [2319](#)
- std::logical_and< void >, [2320](#)
- std::logical_not< _Tp >, [2320](#)
 - argument_type, [2321](#)
 - result_type, [2321](#)
- std::logical_not< void >, [2321](#)
- std::logical_or< _Tp >, [2321](#)
 - first_argument_type, [2322](#)

result_type, 2322
 second_argument_type, 2322
 std::logical_or< void >, 2322
 std::lognormal_distribution< _RealType >, 2323
 max, 2324
 min, 2324
 operator<=, 2325
 operator>=, 2325
 operator(), 2324
 operator==, 2325
 param, 2324
 reset, 2324
 result_type, 2324
 std::lognormal_distribution< _RealType >::param_type, 2603
 std::make_signed< _Tp >, 2332
 std::make_unsigned< _Tp >, 2332
 std::map< _Key, _Tp, _Compare, _Alloc >, 2336
 ~map, 2342
 at, 2343
 begin, 2343
 cbegin, 2343
 cend, 2344
 clear, 2344
 contains, 2344
 count, 2345
 crbegin, 2345
 crend, 2345
 emplace, 2346
 emplace_hint, 2346
 empty, 2347
 end, 2347
 equal_range, 2347, 2348
 erase, 2349, 2350
 extract, 2350, 2351
 find, 2351, 2352
 get_allocator, 2352
 insert, 2352–2356
 insert_or_assign, 2356, 2357
 key_comp, 2357
 lower_bound, 2357–2359
 map, 2340–2342
 max_size, 2359
 operator=, 2359, 2360
 operator[], 2360
 rbegin, 2360
 rend, 2361
 size, 2361
 swap, 2361
 try_emplace, 2361, 2362
 upper_bound, 2362–2364
 value_comp, 2364
 std::mask_array< _Tp >, 2364
 operator<=, 2366
 operator>=, 2366
 operator*=, 2366
 operator^=, 2366
 operator+=, 2366
 operator-=, 2366
 operator/=: 2366
 operator%=: 2365
 operator&=: 2366
 operator|=, 2367
 std::match_results< _Bi_iter, _Alloc >, 2368
 ~match_results, 2370
 begin, 2371
 cbegin, 2371
 cend, 2371
 empty, 2371
 end, 2371
 format, 2371, 2372
 get_allocator, 2372
 length, 2372
 match_results, 2370
 max_size, 2373
 operator=, 2373
 operator[], 2373
 position, 2374
 prefix, 2374
 ready, 2374
 size, 2375
 str, 2375
 suffix, 2375
 swap, 2375
 std::mem_fun1_ref_t< _Ret, _Tp, _Arg >, 2377
 first_argument_type, 2377
 result_type, 2377
 second_argument_type, 2378
 std::mem_fun1_t< _Ret, _Tp, _Arg >, 2378
 first_argument_type, 2379
 result_type, 2379
 second_argument_type, 2379
 std::mem_fun_ref_t< _Ret, _Tp >, 2379
 argument_type, 2380
 result_type, 2380
 std::mem_fun_t< _Ret, _Tp >, 2380
 argument_type, 2381
 result_type, 2381
 std::mergeable, 794
 std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >, 2381
 discard, 2384
 max, 2384
 mersenne_twister_engine, 2384
 min, 2384
 operator<=, 2384
 operator>=, 2385

- operator==, 2385
- result_type, 2383
- std::messages< _CharT >, 2386
 - ~messages, 2388
 - char_type, 2388
 - do_get, 2389
 - id, 2389
 - messages, 2388
 - string_type, 2388
- std::messages_base, 2389
- std::messages_byname< _CharT >, 2390
 - do_get, 2391
 - id, 2391
- std::minus< _Tp >, 2391
 - first_argument_type, 2392
 - result_type, 2392
 - second_argument_type, 2392
- std::minus< void >, 2393
- std::modulus< _Tp >, 2394
 - first_argument_type, 2394
 - result_type, 2394
 - second_argument_type, 2395
- std::modulus< void >, 2395
- std::money_base, 2395
- std::money_get< _CharT, _InIter >, 2397
 - ~money_get, 2398
 - char_type, 2398
 - do_get, 2399
 - get, 2399, 2400
 - id, 2401
 - iter_type, 2398
 - money_get, 2398
 - string_type, 2398
- std::money_put< _CharT, _OutIter >, 2401
 - ~money_put, 2403
 - char_type, 2402
 - do_put, 2403
 - id, 2405
 - iter_type, 2402
 - money_put, 2402
 - put, 2404
 - string_type, 2402
- std::moneypunct< _CharT, _Intl >, 2405
 - ~moneypunct, 2408
 - char_type, 2407
 - curr_symbol, 2409
 - decimal_point, 2409
 - do_curr_symbol, 2409
 - do_decimal_point, 2409
 - do_frac_digits, 2409
 - do_grouping, 2410
 - do_neg_format, 2410
 - do_negative_sign, 2410
 - do_pos_format, 2411
 - do_positive_sign, 2411
 - do_thousands_sep, 2411
 - frac_digits, 2412
 - grouping, 2412
 - id, 2414
 - intl, 2414
 - moneypunct, 2408
 - neg_format, 2412
 - negative_sign, 2413
 - pos_format, 2413
 - positive_sign, 2413
 - string_type, 2407
 - thousands_sep, 2414
- std::moneypunct_byname< _CharT, _Intl >, 2414
 - curr_symbol, 2416
 - decimal_point, 2416
 - do_curr_symbol, 2417
 - do_decimal_point, 2417
 - do_frac_digits, 2417
 - do_grouping, 2418
 - do_neg_format, 2418
 - do_negative_sign, 2418
 - do_pos_format, 2418
 - do_positive_sign, 2419
 - do_thousands_sep, 2419
 - frac_digits, 2419
 - grouping, 2420
 - id, 2422
 - neg_format, 2420
 - negative_sign, 2420
 - pos_format, 2421
 - positive_sign, 2421
 - thousands_sep, 2421
- std::move_constructible, 795
- std::move_iterator< _Iterator >, 2422
- std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV
noexcept(_Noex)>, 2423
 - move_only_function, 2424, 2425
 - operator bool, 2425
 - operator(), 2425
 - operator=, 2425, 2426
 - operator==, 2426
 - swap, 2426
- std::multimap< _Key, _Tp, _Compare, _Alloc >, 2429
 - ~multimap, 2437
 - begin, 2437
 - cbegin, 2437
 - cend, 2437
 - clear, 2437
 - contains, 2437, 2438
 - count, 2438
 - crbegin, 2439
 - crend, 2439
 - emplace, 2439

- emplace_hint, 2439
- empty, 2440
- end, 2440
- equal_range, 2440–2442
- erase, 2442, 2443
- extract, 2444
- find, 2444, 2445
- get_allocator, 2445
- insert, 2446–2449
- key_comp, 2449
- lower_bound, 2449–2451
- max_size, 2451
- multimap, 2434–2436
- operator=, 2451, 2452
- rbegin, 2452
- rend, 2452
- size, 2452
- swap, 2453
- upper_bound, 2453, 2454
- value_comp, 2454
- std::multiplies< _Tp >, 2455
 - first_argument_type, 2455
 - result_type, 2455
 - second_argument_type, 2455
- std::multiplies< void >, 2456
- std::multiset< _Key, _Compare, _Alloc >, 2458
 - ~multiset, 2464
 - begin, 2464
 - cbegin, 2465
 - cend, 2465
 - clear, 2465
 - contains, 2465
 - count, 2466
 - crbegin, 2466
 - crend, 2466
 - emplace, 2467
 - emplace_hint, 2467
 - empty, 2467
 - end, 2468
 - equal_range, 2468, 2469
 - erase, 2469, 2470
 - extract, 2471
 - find, 2471, 2472
 - get_allocator, 2472
 - insert, 2473, 2474
 - key_comp, 2474
 - lower_bound, 2475, 2476
 - max_size, 2476
 - multiset, 2462–2464
 - operator=, 2476, 2477
 - rbegin, 2477
 - rend, 2477
 - size, 2477
 - swap, 2477
 - upper_bound, 2478, 2480
 - value_comp, 2480
- std::mutex, 2484
- std::negate< _Tp >, 2484
 - argument_type, 2485
 - result_type, 2485
- std::negate< void >, 2485
- std::negative_binomial_distribution< _IntType >, 2485
 - k, 2487
 - max, 2487
 - min, 2487
 - operator<=, 2488
 - operator>, 2488
 - operator(), 2487
 - operator==, 2488
 - p, 2487
 - param, 2487
 - reset, 2487
 - result_type, 2486
- std::negative_binomial_distribution< _IntType >::param_type, 2604
- std::nested_exception, 2489
- std::normal_distribution< _RealType >, 2491
 - max, 2493
 - mean, 2493
 - min, 2493
 - normal_distribution, 2493
 - operator<=, 2494
 - operator>, 2495
 - operator(), 2493
 - operator==, 2494
 - param, 2493, 2494
 - reset, 2494
 - result_type, 2492
 - stddev, 2494
- std::normal_distribution< _RealType >::param_type, 2604
- std::nostream_t, 2495
- std::not_equal_to< _Tp >, 2495
 - first_argument_type, 2496
 - result_type, 2496
 - second_argument_type, 2496
- std::not_equal_to< void >, 2497
- std::nullopt_t, 2499
- std::num_get< _CharT, _InIter >, 2499
 - ~num_get, 2502
 - char_type, 2502
 - do_get, 2502–2508
 - get, 2508–2514
 - id, 2515
 - iter_type, 2502
 - num_get, 2502
- std::num_put< _CharT, _OutIter >, 2515
 - ~num_put, 2518

- char_type, [2517](#)
- do_put, [2518–2522](#)
- id, [2528](#)
- iter_type, [2517](#)
- num_put, [2518](#)
- put, [2522–2527](#)
- std::numbers, [771](#)
 - e_v, [772](#)
 - egamma_v, [772](#)
 - inv_pi_v, [772](#)
 - inv_sqrt3_v, [772](#)
 - inv_sqrtpi_v, [772](#)
 - ln10_v, [772](#)
 - ln2_v, [772](#)
 - log10e_v, [773](#)
 - log2e_v, [773](#)
 - phi_v, [773](#)
 - pi_v, [773](#)
 - sqrt2_v, [773](#)
 - sqrt3_v, [773](#)
- std::numeric_limits< _Tp >, [2528](#)
 - denorm_min, [2529](#)
 - digits, [2531](#)
 - digits10, [2531](#)
 - epsilon, [2529](#)
 - has_denorm, [2531](#)
 - has_denorm_loss, [2531](#)
 - has_infinity, [2531](#)
 - has_quiet_NaN, [2531](#)
 - has_signaling_NaN, [2531](#)
 - infinity, [2530](#)
 - is_bounded, [2531](#)
 - is_exact, [2532](#)
 - is_iec559, [2532](#)
 - is_integer, [2532](#)
 - is_modulo, [2532](#)
 - is_signed, [2532](#)
 - is_specialized, [2532](#)
 - lowest, [2530](#)
 - max, [2530](#)
 - max_digits10, [2532](#)
 - max_exponent, [2532](#)
 - max_exponent10, [2533](#)
 - min, [2530](#)
 - min_exponent, [2533](#)
 - min_exponent10, [2533](#)
 - quiet_NaN, [2530](#)
 - radix, [2533](#)
 - round_error, [2530](#)
 - round_style, [2533](#)
 - signaling_NaN, [2531](#)
 - tinyness_before, [2533](#)
 - traps, [2533](#)
- std::numeric_limits< bool >, [2534](#)
- std::numeric_limits< char >, [2535](#)
- std::numeric_limits< char16_t >, [2536](#)
- std::numeric_limits< char32_t >, [2537](#)
- std::numeric_limits< double >, [2538](#)
- std::numeric_limits< float >, [2539](#)
- std::numeric_limits< int >, [2540](#)
- std::numeric_limits< long >, [2541](#)
- std::numeric_limits< long double >, [2542](#)
- std::numeric_limits< long long >, [2543](#)
- std::numeric_limits< short >, [2544](#)
- std::numeric_limits< signed char >, [2545](#)
- std::numeric_limits< unsigned char >, [2546](#)
- std::numeric_limits< unsigned int >, [2547](#)
- std::numeric_limits< unsigned long >, [2548](#)
- std::numeric_limits< unsigned long long >, [2549](#)
- std::numeric_limits< unsigned short >, [2550](#)
- std::numeric_limits< wchar_t >, [2551](#)
- std::numpunct< _CharT >, [2552](#)
 - ~numpunct, [2554](#)
 - char_type, [2553](#)
 - decimal_point, [2554](#)
 - do_decimal_point, [2554](#)
 - do_falsename, [2555](#)
 - do_grouping, [2555](#)
 - do_thousands_sep, [2555](#)
 - do_truename, [2555](#)
 - falsename, [2556](#)
 - grouping, [2556](#)
 - id, [2557](#)
 - numpunct, [2553, 2554](#)
 - string_type, [2553](#)
 - thousands_sep, [2556](#)
 - truename, [2556](#)
- std::numpunct_byname< _CharT >, [2557](#)
 - decimal_point, [2558](#)
 - do_decimal_point, [2559](#)
 - do_falsename, [2559](#)
 - do_grouping, [2559](#)
 - do_thousands_sep, [2559](#)
 - do_truename, [2560](#)
 - falsename, [2560](#)
 - grouping, [2560](#)
 - id, [2561](#)
 - thousands_sep, [2560](#)
 - truename, [2561](#)
- std::once_flag, [2563](#)
 - call_once, [2564](#)
 - once_flag, [2563](#)
 - operator=, [2564](#)
- std::optional< _Tp >, [2566](#)
- std::ostream_iterator< _Tp, _CharT, _Traits >, [2568](#)
 - char_type, [2569](#)
 - difference_type, [2569](#)
 - iterator_category, [2569](#)

- operator=, [2570](#)
- ostream_iterator, [2570](#)
- ostream_type, [2569](#)
- pointer, [2569](#)
- reference, [2569](#)
- traits_type, [2569](#)
- value_type, [2570](#)
- std::ostreambuf_iterator< _CharT, _Traits >, [2571](#)
 - char_type, [2573](#)
 - difference_type, [2573](#)
 - failed, [2574](#)
 - iterator_category, [2573](#)
 - operator*, [2574](#)
 - operator++, [2574](#)
 - operator=, [2574](#)
 - ostream_type, [2573](#)
 - ostreambuf_iterator, [2574](#)
 - pointer, [2573](#)
 - reference, [2573](#)
 - streambuf_type, [2573](#)
 - traits_type, [2573](#)
 - value_type, [2573](#)
- std::out_of_range, [2575](#)
 - what, [2575](#)
- std::output_iterator_tag, [2575](#)
- std::overflow_error, [2584](#)
 - what, [2584](#)
- std::owner_less< _Tp >, [2585](#)
- std::owner_less< shared_ptr< _Tp > >, [2586](#)
 - first_argument_type, [2586](#)
 - result_type, [2586](#)
 - second_argument_type, [2586](#)
- std::owner_less< void >, [2586](#)
 - first_argument_type, [2587](#)
 - result_type, [2587](#)
 - second_argument_type, [2587](#)
- std::owner_less< weak_ptr< _Tp > >, [2588](#)
 - first_argument_type, [2588](#)
 - result_type, [2588](#)
 - second_argument_type, [2588](#)
- std::packaged_task< _Res(_ArgTypes...)>, [2589](#)
- std::pair< _T1, _T2 >, [2589](#)
 - first, [2593](#)
 - first_type, [2592](#)
 - pair, [2592](#)
 - second, [2593](#)
 - second_type, [2592](#)
 - swap, [2593](#)
- std::permutable, [795](#)
- std::piecewise_constant_distribution< _RealType >, [2620](#)
 - densities, [2621](#)
 - intervals, [2621](#)
 - max, [2621](#)
 - min, [2621](#)
- operator<<, [2622](#)
- operator>>, [2623](#)
- operator(), [2622](#)
- operator==, [2623](#)
- param, [2622](#)
- reset, [2622](#)
- result_type, [2621](#)
- std::piecewise_constant_distribution< _RealType >::param_type, [2605](#)
- std::piecewise_construct_t, [2623](#)
- std::piecewise_linear_distribution< _RealType >, [2623](#)
 - densities, [2625](#)
 - intervals, [2625](#)
 - max, [2625](#)
 - min, [2625](#)
 - operator<<, [2626](#)
 - operator>>, [2626](#)
 - operator(), [2625](#)
 - operator==, [2626](#)
 - param, [2625](#)
 - reset, [2626](#)
 - result_type, [2625](#)
- std::piecewise_linear_distribution< _RealType >::param_type, [2606](#)
- std::placeholders, [773](#)
- std::plus< _Tp >, [2627](#)
 - first_argument_type, [2628](#)
 - operator(), [2628](#)
 - result_type, [2628](#)
 - second_argument_type, [2628](#)
- std::pmr::memory_resource, [2381](#)
- std::pmr::pool_options, [2637](#)
 - max_blocks_per_chunk, [2638](#)
- std::pmr::synchronized_pool_resource, [2864](#)
 - do_allocate, [2865](#)
 - do_deallocate, [2865](#)
 - do_is_equal, [2865](#)
- std::pmr::unsynchronized_pool_resource, [3091](#)
 - do_allocate, [3091](#)
 - do_deallocate, [3091](#)
 - do_is_equal, [3092](#)
- std::pointer_to_binary_function< _Arg1, _Arg2, _Result >, [2629](#)
 - first_argument_type, [2630](#)
 - result_type, [2630](#)
 - second_argument_type, [2630](#)
- std::pointer_to_unary_function< _Arg, _Result >, [2630](#)
 - argument_type, [2631](#)
 - result_type, [2631](#)
- std::pointer_traits< _Ptr >, [2631](#)
 - difference_type, [2632](#)
 - element_type, [2632](#)
 - pointer, [2632](#)
 - pointer_to, [2632](#)

- rebind, 2632
- std::pointer_traits< _Tp * >, 2633
 - difference_type, 2633
 - element_type, 2633
 - pointer, 2633
 - pointer_to, 2634
 - rebind, 2633
- std::poisson_distribution< _IntType >, 2634
 - max, 2635
 - mean, 2635
 - min, 2635
 - operator<=, 2636
 - operator>=, 2637
 - operator(), 2635, 2636
 - operator==, 2637
 - param, 2636
 - reset, 2636
 - result_type, 2635
- std::poisson_distribution< _IntType >::param_type, 2606
- std::predicate, 795
- std::priority_queue< _Tp, _Sequence, _Compare >, 2640
 - empty, 2643
 - pop, 2643
 - priority_queue, 2642
 - push, 2643
 - size, 2643
 - top, 2644
- std::projected< _Iter, _Proj >, 2646
- std::promise< _Res >, 2647
- std::promise< _Res & >, 2647
- std::promise< void >, 2648
- std::queue< _Tp, _Sequence >, 2650
 - back, 2652
 - c, 2653
 - empty, 2652
 - front, 2652
 - pop, 2652
 - push, 2652
 - queue, 2652
 - size, 2653
- std::random_access_iterator_tag, 2655
- std::random_device, 2656
 - result_type, 2656
- std::range_error, 2657
 - what, 2657
- std::ranges::bidirectional_range, 795
- std::ranges::borrowed_range, 796
- std::ranges::common_range, 796
- std::ranges::contiguous_range, 796
- std::ranges::dangling, 1948
- std::ranges::empty_view< _Tp >, 2018
- std::ranges::equal_to, 2046
- std::ranges::forward_range, 796
- std::ranges::greater, 2126
- std::ranges::greater_equal, 2128
- std::ranges::input_range, 797
- std::ranges::less, 2273
- std::ranges::less_equal, 2275
- std::ranges::not_equal_to, 2497
- std::ranges::output_range, 797
- std::ranges::random_access_range, 797
- std::ranges::range, 797
- std::ranges::single_view< _Tp >, 2789
- std::ranges::sized_range, 797
- std::ranges::subrange< _It, _Sent, _Kind >, 2857
- std::ranges::view, 798
- std::ranges::view_base, 3124
- std::ranges::view_interface< _Derived >, 3124
- std::ranges::viewable_range, 798
- std::rank< typename >, 2664
- std::ratio< _Num, _Den >, 2664
- std::ratio_equal< _R1, _R2 >, 2665
- std::ratio_greater< _R1, _R2 >, 2666
- std::ratio_greater_equal< _R1, _R2 >, 2667
- std::ratio_less< _R1, _R2 >, 2667
- std::ratio_less_equal< _R1, _R2 >, 2668
- std::ratio_not_equal< _R1, _R2 >, 2669
- std::raw_storage_iterator< _OutputIterator, _Tp >, 2669
 - difference_type, 2670
 - iterator_category, 2670
 - pointer, 2671
 - reference, 2671
 - value_type, 2671
- std::recursive_mutex, 2687
- std::recursive_timed_mutex, 2687
- std::reference_wrapper< _Tp >, 2688
 - cref, 2689
 - ref, 2689, 2690
- std::regex_constants, 774
 - __multiline, 780
 - __polynomial, 780
 - awk, 780
 - basic, 780
 - collate, 781
 - ECMAScript, 781
 - egrep, 781
 - error_backref, 776
 - error_badbrace, 776
 - error_badrepeat, 776
 - error_brace, 776
 - error_brack, 776
 - error_collate, 777
 - error_complexity, 777
 - error_ctype, 777
 - error_escape, 777
 - error_paren, 777
 - error_range, 777
 - error_space, 777

- error_stack, 777
- error_type, 776
- extended, 781
- format_default, 781
- format_first_only, 781
- format_no_copy, 782
- format_sed, 782
- grep, 782
- icase, 782
- match_any, 782
- match_continuous, 782
- match_default, 782
- match_flag_type, 776
- match_not_bol, 782
- match_not_bow, 782
- match_not_eol, 782
- match_not_eow, 783
- match_not_null, 783
- match_prev_avail, 783
- multiline, 783
- nosubs, 783
- operator~, 780
- operator^, 778
- operator^=, 779
- operator&, 777, 778
- operator&=, 778
- operator|, 779
- operator|=, 779
- optimize, 783
- syntax_option_type, 776
- std::regex_error, 2690
 - code, 2691
 - regex_error, 2691
 - what, 2691
- std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2691
 - operator*, 2693
 - operator++, 2693
 - operator->, 2693
 - operator=, 2693
 - operator==, 2693
 - regex_iterator, 2692
- std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 2694
 - operator*, 2697
 - operator++, 2697
 - operator->, 2697
 - operator=, 2697
 - operator==, 2698
 - regex_token_iterator, 2695, 2696
- std::regex_traits< _Ch_type >, 2698
 - getloc, 2699
 - imbue, 2699
 - isctype, 2699
 - length, 2700
 - lookup_classname, 2700
 - lookup_collatename, 2701
 - regex_traits, 2699
 - transform, 2701
 - transform_primary, 2702
 - translate, 2702
 - translate_nocase, 2703
 - value, 2703
- std::regular_invocable, 798
- std::rel_ops, 783
 - operator!=, 784
 - operator<=, 784
 - operator>, 785
 - operator>=, 785
- std::relation, 798
- std::remove_all_extents< _Tp >, 2704
- std::remove_const< _Tp >, 2704
- std::remove_cv< _Tp >, 2704
- std::remove_extent< _Tp >, 2705
- std::remove_pointer< _Tp >, 2705
- std::remove_reference< _Tp >, 2705
- std::remove_volatile< _Tp >, 2705
- std::result_of< _Signature >, 2707
- std::reverse_iterator< _Iterator >, 2708
 - base, 2710
 - operator*, 2710
 - operator+, 2710
 - operator++, 2710
 - operator+=, 2711
 - operator-, 2711
 - operator->, 2712
 - operator--, 2711
 - operator=, 2711
 - operator[], 2712
 - reverse_iterator, 2709
- std::runtime_error, 2717
 - runtime_error, 2718
 - what, 2718
- std::same_as, 799
- std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs >, 2733
- std::scoped_lock< _MutexTypes >, 2734
- std::seed_seq, 2734
 - result_type, 2735
 - seed_seq, 2735
- std::set< _Key, _Compare, _Alloc >, 2743
 - ~set, 2751
 - allocator_type, 2746
 - begin, 2751
 - cbegin, 2751
 - cend, 2751
 - clear, 2751
 - const_iterator, 2746

- const_pointer, 2746
- const_reference, 2746
- const_reverse_iterator, 2747
- contains, 2751, 2752
- count, 2752
- crbegin, 2753
- crend, 2753
- difference_type, 2747
- emplace, 2753
- emplace_hint, 2753
- empty, 2754
- end, 2754
- equal_range, 2754–2756
- erase, 2756, 2757
- extract, 2757
- find, 2757–2759
- get_allocator, 2759
- insert, 2759–2761
- iterator, 2747
- key_comp, 2761
- key_compare, 2747
- key_type, 2747
- lower_bound, 2761, 2762
- max_size, 2763
- operator=, 2763
- pointer, 2747
- rbegin, 2763
- reference, 2747
- rend, 2764
- reverse_iterator, 2747
- set, 2748–2751
- size, 2764
- size_type, 2748
- swap, 2764
- upper_bound, 2764, 2765
- value_comp, 2766
- value_compare, 2748
- value_type, 2748
- std::shared_future< _Res >, 2766
 - _M_get_result, 2767
 - get, 2767
 - shared_future, 2767
- std::shared_future< _Res & >, 2768
 - _M_get_result, 2769
 - get, 2769
 - shared_future, 2769
- std::shared_future< void >, 2770
 - _M_get_result, 2771
 - shared_future, 2770, 2771
- std::shared_lock< _Mutex >, 2771
- std::shared_mutex, 2772
- std::shared_ptr< _Tp >, 2772
 - element_type, 2776
 - get, 2783
 - operator bool, 2783
 - owner_before, 2783
 - shared_ptr, 2777–2782
 - swap, 2783
 - unique, 2783
 - use_count, 2783
 - weak_type, 2776
- std::shared_timed_mutex, 2784
- std::shuffle_order_engine< _RandomNumberEngine, __k >, 2784
 - base, 2787
 - discard, 2787
 - max, 2787
 - min, 2787
 - operator<=, 2788
 - operator>=, 2788
 - operator(), 2787
 - operator==, 2788
 - result_type, 2785
 - seed, 2787, 2788
 - shuffle_order_engine, 2785, 2786
- std::slice, 2790
 - operator==, 2790
- std::slice_array< _Tp >, 2791
 - operator<=, 2793
 - operator>=, 2793
 - operator*=, 2792
 - operator^=, 2793
 - operator+=, 2792
 - operator-=, 2792
 - operator/=, 2792
 - operator%=, 2792
 - operator&=, 2792
 - operator|=, 2793
- std::sortable, 799
- std::source_location, 2795
- std::stack< _Tp, _Sequence >, 2801
 - empty, 2803
 - pop, 2803
 - push, 2803
 - size, 2803
 - stack, 2803
 - top, 2803
- std::stop_callback< _Callback >, 2840
- std::stop_source, 2841
- std::stop_token, 2841
- std::strict_weak_order, 799
- std::student_t_distribution< _RealType >, 2846
 - max, 2847
 - min, 2847
 - operator<=, 2848
 - operator>=, 2849
 - operator(), 2847
 - operator==, 2848

- param, 2848
- reset, 2848
- result_type, 2847
- std::student_t_distribution< _RealType >::param_type, 2607
- std::sub_match< _Biter >, 2849
 - compare, 2853, 2854
 - first, 2857
 - first_type, 2853
 - length, 2854
 - make_pair, 2855
 - operator string_type, 2854
 - operator!=, 2856
 - operator<, 2856
 - operator<=, 2856
 - operator>, 2856
 - operator>=, 2856
 - operator==, 2856
 - second, 2857
 - second_type, 2853
 - str, 2855
 - swap, 2855, 2856
- std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2859
 - discard, 2860
 - max, 2860
 - min, 2860
 - operator<<, 2861
 - operator>>, 2862
 - operator(), 2860
 - operator==, 2861
 - result_type, 2860
 - seed, 2861
 - subtract_with_carry_engine, 2860
- std::system_error, 2866
 - what, 2867
- std::this_thread, 785
 - get_id, 786
 - sleep_for, 786
 - sleep_until, 786
 - yield, 786
- std::thread, 2872
 - native_handle, 2873
- std::thread::id, 2170
- std::time_base, 2881
- std::time_get< _CharT, _InIter >, 2881
 - ~time_get, 2884
 - char_type, 2883
 - date_order, 2884
 - do_date_order, 2884
 - do_get, 2884
 - do_get_date, 2885
 - do_get_monthname, 2886
 - do_get_time, 2886
 - do_get_weekday, 2887
 - do_get_year, 2888
 - get, 2888, 2889
 - get_date, 2889
 - get_monthname, 2890
 - get_time, 2891
 - get_weekday, 2891
 - get_year, 2892
 - id, 2892
 - iter_type, 2883
 - time_get, 2884
- std::time_get_byname< _CharT, _InIter >, 2893
 - date_order, 2894
 - do_date_order, 2895
 - do_get, 2895
 - do_get_date, 2895
 - do_get_monthname, 2896
 - do_get_time, 2897
 - do_get_weekday, 2897
 - do_get_year, 2898
 - get, 2898, 2899
 - get_date, 2900
 - get_monthname, 2900
 - get_time, 2901
 - get_weekday, 2901
 - get_year, 2902
 - id, 2903
- std::time_put< _CharT, _OutIter >, 2904
 - ~time_put, 2905
 - char_type, 2905
 - do_put, 2906
 - id, 2907
 - iter_type, 2905
 - put, 2906, 2907
 - time_put, 2905
- std::time_put_byname< _CharT, _OutIter >, 2908
 - do_put, 2909
 - id, 2911
 - put, 2909, 2910
- std::timed_mutex, 2911
- std::to_chars_result, 2911
- std::tr1, 786
- std::tr1::__detail, 789
- std::tr2, 789
- std::tr2::__detail, 790
- std::tr2::__dynamic_bitset_base< _WordT, _Alloc >, 838
- std::tr2::__M_w, 839
- std::tr2::__reflection_typelist< _Elements >, 873
- std::tr2::__reflection_typelist< _First, _Rest... >, 874
- std::tr2::__reflection_typelist<>, 874
- std::tr2::bases< _Tp >, 1168
- std::tr2::bool_set, 1775
 - bool_set, 1776
 - equals, 1776

- is_emptyset, 1776
- is_indeterminate, 1776
- is_singleton, 1776
- operator bool, 1776
- std::tr2::direct_bases< _Tp >, 1987
- std::tr2::dynamic_bitset< _WordT, _Alloc >, 2004
 - all, 2009
 - any, 2009
 - append, 2009
 - clear, 2009
 - count, 2009
 - dynamic_bitset, 2007, 2008
 - empty, 2009
 - find_first, 2009
 - find_next, 2010
 - flip, 2010
 - get_allocator, 2011
 - max_size, 2011
 - none, 2011
 - num_blocks, 2011
 - operator<<, 2012
 - operator<=, 2012
 - operator>>, 2012
 - operator>=, 2012
 - operator~, 2014
 - operator^=, 2014
 - operator-=, 2012
 - operator=, 2012
 - operator&=, 2011
 - operator[], 2013
 - operator|=, 2014
 - push_back, 2014
 - reset, 2014
 - resize, 2016
 - set, 2016
 - size, 2016
 - swap, 2016
 - test, 2017
 - to_string, 2017
 - to_ullong, 2017
 - to_ulong, 2018
- std::tr2::dynamic_bitset< _WordT, _Alloc >::reference, 2688
- std::try_to_lock_t, 2941
- std::tuple< _Elements >, 2941
- std::tuple< _T1, _T2 >, 2943
- std::tuple_element< 0, pair< _Tp1, _Tp2 > >, 2945
- std::tuple_element< 1, pair< _Tp1, _Tp2 > >, 2946
- std::tuple_element< __i, _Tp >, 2945
- std::tuple_element< __i, tuple< _Types... > >, 2946
- std::tuple_element< _Ind, array< _Tp, _Nm > >, 2946
- std::tuple_size< _Tp >, 2946
- std::tuple_size< array< _Tp, _Nm > >, 2947
- std::tuple_size< pair< _Tp1, _Tp2 > >, 2948
- std::tuple_size< tuple< _Elements... > >, 2949
- std::type_index, 2950
- std::type_info, 2950
 - ~type_info, 2951
 - before, 2951
 - name, 2951
- std::unary_function< _Arg, _Result >, 2954
 - argument_type, 2954
 - result_type, 2954
- std::unary_negate< _Predicate >, 2955
 - argument_type, 2955
 - result_type, 2955
- std::underflow_error, 2957
 - what, 2957
- std::underlying_type< _Tp >, 2957
- std::uniform_int_distribution< _IntType >, 2958
 - max, 2959
 - min, 2959
 - operator(), 2959
 - operator==, 2960
 - param, 2959, 2960
 - reset, 2960
 - result_type, 2959
 - uniform_int_distribution, 2959
- std::uniform_int_distribution< _IntType >::param_type, 2607
- std::uniform_random_bit_generator, 799
- std::uniform_real_distribution< _RealType >, 2960
 - max, 2962
 - min, 2962
 - operator(), 2962
 - operator==, 2963
 - param, 2962
 - reset, 2962
 - result_type, 2961
 - uniform_real_distribution, 2961
- std::uniform_real_distribution< _RealType >::param_type, 2608
- std::unique_lock< _Mutex >, 2963
 - swap, 2964
- std::unique_ptr< _Tp, _Dp >, 2964
 - ~unique_ptr, 2967
 - get, 2967
 - get_deleter, 2968
 - operator bool, 2968
 - operator*, 2968
 - operator->, 2968
 - operator=, 2968, 2969
 - release, 2969
 - reset, 2969
 - swap, 2969
 - unique_ptr, 2966, 2967
- std::unique_ptr< _Tp[], _Dp >, 2970
 - ~unique_ptr, 2972

get, 2972
 get_deleter, 2973
 operator bool, 2973
 operator=, 2973
 operator[], 2974
 release, 2974
 reset, 2974
 swap, 2974
 unique_ptr, 2971, 2972
 std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >, 2978
 allocator_type, 2983
 at, 2987
 begin, 2988
 bucket_count, 2990
 cbegin, 2990
 cend, 2990
 clear, 2991
 const_iterator, 2983
 const_local_iterator, 2983
 const_pointer, 2983
 const_reference, 2983
 contains, 2991
 count, 2992
 difference_type, 2983
 emplace, 2992
 emplace_hint, 2993
 empty, 2993
 end, 2993, 2994
 equal_range, 2994, 2995
 erase, 2996, 2997
 extract, 2997, 2998
 find, 2998, 2999
 get_allocator, 2999
 hash_function, 2999
 hasher, 2984
 insert, 2999–3003
 insert_or_assign, 3003, 3004
 iterator, 2984
 key_eq, 3004
 key_equal, 2984
 key_type, 2984
 load_factor, 3004
 local_iterator, 2984
 mapped_type, 2984
 max_bucket_count, 3004
 max_load_factor, 3004, 3005
 max_size, 3005
 operator=, 3005, 3006
 operator[], 3006
 pointer, 2984
 reference, 2985
 rehash, 3006
 reserve, 3007
 size, 3007
 size_type, 2985
 swap, 3007
 try_emplace, 3007, 3008
 unordered_map, 2985–2987
 value_type, 2985
 std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >, 3011
 allocator_type, 3015
 begin, 3019, 3020
 bucket_count, 3020
 cbegin, 3020, 3021
 cend, 3021
 clear, 3021
 const_iterator, 3015
 const_local_iterator, 3015
 const_pointer, 3015
 const_reference, 3016
 contains, 3022
 count, 3022, 3023
 difference_type, 3016
 emplace, 3023
 emplace_hint, 3023
 empty, 3024
 end, 3024, 3025
 equal_range, 3025, 3026
 erase, 3026, 3027
 extract, 3028
 find, 3028, 3029
 get_allocator, 3030
 hash_function, 3030
 hasher, 3016
 insert, 3030–3033
 iterator, 3016
 key_eq, 3033
 key_equal, 3016
 key_type, 3016
 load_factor, 3033
 local_iterator, 3016
 mapped_type, 3017
 max_bucket_count, 3034
 max_load_factor, 3034
 max_size, 3034
 operator=, 3034, 3035
 pointer, 3017
 reference, 3017
 rehash, 3035
 reserve, 3035
 size, 3036
 size_type, 3017
 swap, 3036
 unordered_multimap, 3017–3019
 value_type, 3017

- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
 >, 3039
 - allocator_type, 3042
 - begin, 3046, 3047
 - bucket_count, 3047
 - cbegin, 3047, 3048
 - cend, 3048
 - clear, 3048
 - const_iterator, 3042
 - const_local_iterator, 3043
 - const_pointer, 3043
 - const_reference, 3043
 - contains, 3049
 - count, 3049, 3050
 - difference_type, 3043
 - emplace, 3050
 - emplace_hint, 3050
 - empty, 3052
 - end, 3052, 3053
 - equal_range, 3053, 3054
 - erase, 3054, 3055
 - extract, 3056
 - find, 3056, 3057
 - get_allocator, 3058
 - hash_function, 3058
 - hasher, 3043
 - insert, 3058–3060
 - iterator, 3043
 - key_eq, 3061
 - key_equal, 3043
 - key_type, 3044
 - load_factor, 3061
 - local_iterator, 3044
 - max_bucket_count, 3061
 - max_load_factor, 3061
 - max_size, 3061
 - operator=, 3061, 3062
 - pointer, 3044
 - reference, 3044
 - rehash, 3062
 - reserve, 3062
 - size, 3063
 - size_type, 3044
 - swap, 3063
 - unordered_multiset, 3044–3046
 - value_type, 3044
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, 3066
 - allocator_type, 3070
 - begin, 3074
 - bucket_count, 3074
 - cbegin, 3075
 - cend, 3075
 - clear, 3075
 - const_iterator, 3070
 - const_local_iterator, 3070
 - const_pointer, 3070
 - const_reference, 3070
 - contains, 3076
 - count, 3076, 3077
 - difference_type, 3070
 - emplace, 3077
 - emplace_hint, 3077
 - empty, 3078
 - end, 3078, 3080
 - equal_range, 3080, 3081
 - erase, 3082, 3083
 - extract, 3083
 - find, 3083–3085
 - get_allocator, 3085
 - hash_function, 3085
 - hasher, 3070
 - insert, 3085–3088
 - iterator, 3070
 - key_eq, 3088
 - key_equal, 3071
 - key_type, 3071
 - load_factor, 3088
 - local_iterator, 3071
 - max_bucket_count, 3088
 - max_load_factor, 3088
 - max_size, 3089
 - operator=, 3089
 - pointer, 3071
 - reference, 3071
 - rehash, 3090
 - reserve, 3090
 - size, 3090
 - size_type, 3071
 - swap, 3090
 - unordered_set, 3072, 3073
 - value_type, 3071
- std::unwrap_ref_decay< _Tp >, 3092
- std::unwrap_reference< _Tp >, 3092
- std::uses_allocator< tuple< _Types... >, _Alloc >, 3093
- std::uses_allocator< typename, typename >, 3093
- std::valarray< _Tp >, 3094
 - operator!, 3096
 - operator<=, 3098
 - operator>=, 3098
 - operator*=, 3097
 - operator~, 3099
 - operator^=, 3098
 - operator+, 3097
 - operator+=, 3097
 - operator-, 3097
 - operator-=, 3097
 - operator/=: 3098

- operator%[=](#), [3096](#)
- operator&[=](#), [3096](#)
- operator|[=](#), [3099](#)
- valarray, [3096](#)
- std::vector<_Tp, _Alloc>, [3102](#)
 - _M_allocate_and_copy, [3108](#)
 - _M_range_check, [3108](#)
 - ~vector, [3108](#)
 - assign, [3108](#), [3109](#)
 - at, [3109](#), [3110](#)
 - back, [3110](#)
 - begin, [3111](#)
 - capacity, [3111](#)
 - cbegin, [3111](#)
 - cend, [3111](#)
 - clear, [3111](#)
 - crbegin, [3111](#)
 - crend, [3112](#)
 - data, [3112](#)
 - emplace, [3112](#)
 - empty, [3112](#)
 - end, [3113](#)
 - erase, [3113](#)
 - front, [3114](#)
 - get_allocator, [3114](#)
 - insert, [3114](#)–[3116](#)
 - max_size, [3116](#)
 - operator=, [3116](#), [3117](#)
 - operator[], [3117](#), [3118](#)
 - pop_back, [3118](#)
 - push_back, [3118](#)
 - rbegin, [3118](#), [3119](#)
 - rend, [3119](#)
 - reserve, [3119](#)
 - resize, [3119](#), [3120](#)
 - shrink_to_fit, [3120](#)
 - size, [3120](#)
 - swap, [3120](#)
 - vector, [3105](#)–[3107](#)
- std::vector<bool, _Alloc>, [3121](#)
- std::wbuffer_convert<_Codecvt, _Elem, _Tr>, [3125](#)
 - _M_buf_locale, [3140](#)
 - _M_in_beg, [3140](#)
 - _M_in_cur, [3141](#)
 - _M_in_end, [3141](#)
 - _M_out_beg, [3141](#)
 - _M_out_cur, [3141](#)
 - _M_out_end, [3141](#)
 - __streambuf_type, [3127](#)
 - char_type, [3127](#)
 - eback, [3129](#)
 - egptr, [3129](#)
 - epptr, [3129](#)
 - gbump, [3129](#)
 - getloc, [3130](#)
 - gptr, [3130](#)
 - imbue, [3130](#)
 - in_avail, [3130](#)
 - int_type, [3127](#)
 - off_type, [3128](#)
 - overflow, [3131](#)
 - pbackfail, [3131](#)
 - pbase, [3132](#)
 - pbump, [3132](#)
 - pos_type, [3128](#)
 - pptr, [3132](#)
 - pubimbue, [3132](#)
 - pubseekoff, [3133](#)
 - pubseekpos, [3133](#)
 - pubsetbuf, [3133](#)
 - pubsync, [3133](#)
 - sbumpc, [3134](#)
 - seekoff, [3134](#)
 - seekpos, [3134](#)
 - setbuf, [3134](#)
 - setg, [3135](#)
 - setp, [3135](#)
 - sgetc, [3136](#)
 - sgetn, [3136](#)
 - showmanyc, [3136](#)
 - snextc, [3137](#)
 - sputbackc, [3137](#)
 - sputc, [3137](#)
 - sputn, [3138](#)
 - state, [3138](#)
 - sungetc, [3138](#)
 - sync, [3138](#)
 - traits_type, [3128](#)
 - uflow, [3139](#)
 - underflow, [3139](#)
 - wbuffer_convert, [3128](#)
 - xsgetn, [3139](#)
 - xspn, [3140](#)
- std::weak_ptr<_Tp>, [3141](#)
- std::weakly_incrementable, [799](#)
- std::weibull_distribution<_RealType>, [3142](#)
 - a, [3143](#)
 - b, [3143](#)
 - max, [3144](#)
 - min, [3144](#)
 - operator(), [3144](#)
 - operator==, [3145](#)
 - param, [3144](#)
 - reset, [3144](#)
 - result_type, [3143](#)
- std::weibull_distribution<_RealType>::param_type, [2608](#)

- std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >, 3145
 - converted, 3147
 - from_bytes, 3147
 - state, 3147
 - to_bytes, 3148
 - wstring_convert, 3146
- std_abs.h, 3269
- std_function.h, 3270
- std_mutex.h, 3270
- std_thread.h, 3271
- stdatomic.h, 3541
- stdc++.h, 3593
- stddev
 - std::normal_distribution< _RealType >, 2494
- stdexcept, 3541
- stdio_filebuf
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2807
- stdio_filebuf.h, 3455
- stdio_sync_filebuf.h, 3455
- stdlib.h, 3542
- stdtr1c++.h, 3593
- stl_algo.h, 3271
- stl_algobase.h, 3281
- stl_bvector.h, 3286
- stl_construct.h, 3287
- stl_deque.h, 3288
 - _GLIBCXX_DEQUE_BUF_SIZE, 3288
- stl_function.h, 3289
- stl_heap.h, 3290
- stl_iterator.h, 3292, 3296
- stl_iterator_base_funcs.h, 3297
- stl_iterator_base_types.h, 3298
- stl_list.h, 3298
- stl_map.h, 3299
- stl_multimap.h, 3300
- stl_multiset.h, 3301
- stl_numeric.h, 3302
- stl_pair.h, 3303
 - __cpp_lib_tuples_by_type, 3304
- stl_queue.h, 3304
- stl_raw_storage_iter.h, 3305
- stl_relops.h, 3306
- stl_set.h, 3306
- stl_stack.h, 3307
- stl_tempbuf.h, 3308
- stl_tree.h, 3308
- stl_uninitialized.h, 3309
- stl_vector.h, 3309
- stop_token, 3542
- str
 - std::basic_istream< _CharT, _Traits, _Alloc >, 1422, 1423
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1521
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1673
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1716
- std::match_results< _Bi_iter, _Alloc >, 2375
- std::sub_match< _Biliter >, 2855
- stream_iterator.h, 3310
- streambuf, 3542
 - I/O, 171
- streambuf.tcc, 3311
- streambuf_iterator.h, 3311
- streambuf_type
 - std::istreambuf_iterator< _CharT, _Traits >, 2254
 - std::ostreambuf_iterator< _CharT, _Traits >, 2573
- streamoff
 - std, 633
- streampos
 - std, 633
- streamsize
 - std, 634
- stride
 - Numeric Arrays, 241
- string, 3543, 3545, 3546
 - Strings, 302
- string_conversions.h, 3456
- string_type
 - std::collate< _CharT >, 1839
 - std::collate_byname< _CharT >, 1845
 - std::messages< _CharT >, 2388
 - std::money_get< _CharT, _Inlter >, 2398
 - std::money_put< _CharT, _Outlter >, 2402
 - std::moneypunct< _CharT, _Intl >, 2407
 - std::numpunct< _CharT >, 2553
- string_view, 3546, 3548
- string_view.tcc, 3312
- stringbuf
 - I/O, 171
- stringfwd.h, 3312
- Strings, 302
 - string, 302
 - u16string, 302
 - u32string, 302
 - wstring, 302
- stringstream
 - I/O, 172
- strstream, 3167
- substr
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, 932
 - std::basic_string< _CharT, _Traits, _Alloc >, 1655
- subtract_with_carry_engine
 - std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 2860

- subtractive_rng
 - __gnu_cxx::subtractive_rng, [2863](#)
- suffix
 - std::match_results< _Bi_iter, _Alloc >, [2375](#)
- sum
 - Numeric Arrays, [241](#)
- sungetc
 - __gnu_cxx::enc_filebuf< _CharT >, [2035](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2820](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2836](#)
 - std::basic_filebuf< _CharT, _Traits >, [1185](#)
 - std::basic_streambuf< _CharT, _Traits >, [1552](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1674](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3138](#)
- swap
 - __gnu_cxx, [449](#)
 - __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >, [932](#)
 - __gnu_debug::basic_string< _CharT, _Traits, _Allocator >, [1589](#)
 - __gnu_parallel::__iteratorPair< _Iterator1, _Iterator2, _IteratorCategory >, [967](#), [969](#)
 - __gnu_pbds::sample_probe_fn, [2719](#)
 - __gnu_pbds::sample_range_hashing, [2720](#)
 - __gnu_pbds::sample_ranged_hash_fn, [2721](#)
 - __gnu_pbds::sample_resize_policy, [2724](#)
 - __gnu_pbds::sample_resize_trigger, [2727](#)
 - __gnu_pbds::sample_size_policy, [2728](#)
 - __gnu_pbds::sample_update_policy, [2731](#)
 - Futures, [122](#)
 - Metaprogramming, [400](#), [401](#)
 - Mutexes, [124](#)
 - Numeric Arrays, [241](#)
 - Pointer Abstractions, [386](#)
 - Regular Expressions, [300](#)
 - std, [706–709](#)
 - std::__debug, [720](#)
 - std::any, [1114](#)
 - std::basic_regex< _Ch_type, _Rx_traits >, [1538](#)
 - std::basic_string< _CharT, _Traits, _Alloc >, [1656](#)
 - std::deque< _Tp, _Alloc >, [1986](#)
 - std::experimental::fundamentals_v1::any, [1117](#)
 - std::forward_list< _Tp, _Alloc >, [2090](#)
 - std::function< _Res(_ArgTypes...)>, [2102](#)
 - std::list< _Tp, _Alloc >, [2305](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2361](#)
 - std::match_results< _Bi_iter, _Alloc >, [2375](#)
 - std::move_only_function< _Res(_ArgTypes...) _GLIBCXX_MOF_CV noexcept(_Noex)>, [2426](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2453](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2477](#)
 - std::pair< _T1, _T2 >, [2593](#)
 - std::set< _Key, _Compare, _Alloc >, [2764](#)
 - std::shared_ptr< _Tp >, [2783](#)
 - std::sub_match< _Biter >, [2855](#), [2856](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2016](#)
 - std::unique_lock< _Mutex >, [2964](#)
 - std::unique_ptr< _Tp, _Dp >, [2969](#)
 - std::unique_ptr< _Tp[], _Dp >, [2974](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3007](#)
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3036](#)
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3063](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3090](#)
 - std::vector< _Tp, _Alloc >, [3120](#)
 - Type-safe container of any type, [319](#)
 - Utilities, [348](#), [349](#)
- swap_ranges
 - Mutating, [38](#)
- sync
 - __gnu_cxx::enc_filebuf< _CharT >, [2035](#)
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, [2820](#)
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, [2837](#)
 - std::basic_filebuf< _CharT, _Traits >, [1186](#)
 - std::basic_fstream< _CharT, _Traits >, [1231](#)
 - std::basic_ifstream< _CharT, _Traits >, [1275](#)
 - std::basic_iostream< _CharT, _Traits >, [1343](#)
 - std::basic_istream< _CharT, _Traits >, [1382](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1423](#)
 - std::basic_streambuf< _CharT, _Traits >, [1553](#)
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, [1674](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1716](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3138](#)
- sync_with_stdio
 - std::basic_fstream< _CharT, _Traits >, [1231](#)
 - std::basic_ifstream< _CharT, _Traits >, [1275](#)
 - std::basic_ios< _CharT, _Traits >, [1299](#)
 - std::basic_iostream< _CharT, _Traits >, [1344](#)
 - std::basic_istream< _CharT, _Traits >, [1382](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1423](#)
 - std::basic_ofstream< _CharT, _Traits >, [1457](#)
 - std::basic_ostream< _CharT, _Traits >, [1489](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1522](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1717](#)
 - std::ios_base, [2196](#)
- syncstream, [3550](#)
- syntax_option_type

- std::regex_constants, 776
- synth_access_traits
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >, 2939
 - __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >, 2940
- synth_access_traits.hpp, 3435
- system_category
 - Diagnostics, 134
- system_error, 3550, 3551
- t
 - std::binomial_distribution< _IntType >, 1756
- table
 - std::ctype< char >, 1907
 - std::ctype_byname< char >, 1945
- table_size
 - std::ctype< char >, 1909
 - std::ctype_byname< char >, 1948
- tag_and_trait.hpp, 3445
- Tags, 151
 - trivial_iterator_difference_type, 152
- tags.h, 3528
- tan
 - Complex Numbers, 196
- tanh
 - Complex Numbers, 196
- target
 - std::function< _Res(_ArgTypes...)>, 2102
- target_type
 - std::function< _Res(_ArgTypes...)>, 2103
- Technical Specifications, 303
- tellg
 - std::basic_fstream< _CharT, _Traits >, 1232
 - std::basic_ifstream< _CharT, _Traits >, 1276
 - std::basic_iostream< _CharT, _Traits >, 1344
 - std::basic_istream< _CharT, _Traits >, 1382
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1424
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1717
- tellp
 - std::basic_fstream< _CharT, _Traits >, 1232
 - std::basic_iostream< _CharT, _Traits >, 1344
 - std::basic_ofstream< _CharT, _Traits >, 1458
 - std::basic_ostream< _CharT, _Traits >, 1490
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1522
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1717
- temporary_buffer
 - __gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >, 2868
- terminate
 - Exceptions, 138
- terminate_handler
 - Exceptions, 136
- test
 - std::bitset< _Nb >, 1774
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, 2017
- tgmath.h, 3551
- thin_heap.hpp, 3439
- this_thread_sleep.h, 3313
- thousands_sep
 - std::moneypunct< _CharT, _Intl >, 2414
 - std::moneypunct_byname< _CharT, _Intl >, 2421
 - std::numpunct< _CharT >, 2556
 - std::numpunct_byname< _CharT >, 2560
- thread, 3551
- Threads, 125
- throw_allocator.h, 3456
- throw_with_nested
 - Exceptions, 139
- tie
 - std::basic_fstream< _CharT, _Traits >, 1232
 - std::basic_ifstream< _CharT, _Traits >, 1276
 - std::basic_ios< _CharT, _Traits >, 1300
 - std::basic_iostream< _CharT, _Traits >, 1345
 - std::basic_istream< _CharT, _Traits >, 1383
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1424
 - std::basic_ofstream< _CharT, _Traits >, 1458
 - std::basic_ostream< _CharT, _Traits >, 1490
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, 1522
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1718
 - Utilities, 349
- Time, 408
 - abs, 414
 - ceil, 414, 415
 - days, 413
 - duration_cast, 415
 - floor, 416
 - high_resolution_clock, 413
 - hours, 413
 - microseconds, 413
 - milliseconds, 413
 - minutes, 413
 - months, 413
 - nanoseconds, 413
 - operator!=, 417
 - operator<, 421
 - operator<=, 421
 - operator>, 421

- operator>=, [421](#)
- operator*, [419](#)
- operator+, [419](#), [420](#)
- operator-, [420](#)
- operator/, [420](#)
- operator==, [421](#)
- operator""d, [417](#)
- operator""h, [417](#)
- operator""min, [417](#)
- operator""ms, [418](#)
- operator""ns, [418](#)
- operator""s, [418](#)
- operator""us, [418](#)
- operator""y, [418](#)
- operator%, [419](#)
- round, [421](#), [423](#)
- seconds, [413](#)
- time_point_cast, [423](#)
- weeks, [413](#)
- years, [413](#)
- time
 - std::locale, [2316](#)
- time_get
 - std::time_get< _CharT, _InIter >, [2884](#)
- time_members.h, [3593](#)
- time_point_cast
 - Time, [423](#)
- time_put
 - std::time_put< _CharT, _OutIter >, [2905](#)
- tinyness_before
 - std::__numeric_limits_base, [866](#)
 - std::numeric_limits< _Tp >, [2533](#)
- TLB_size
 - __gnu_parallel::_Settings, [1078](#)
- to_address
 - Pointer Abstractions, [386](#), [387](#)
- to_array
 - Array creation functions, [311](#)
- to_bytes
 - std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >, [3148](#)
- to_string
 - std::bitset< _Nb >, [1774](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2017](#)
- to_ullong
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2017](#)
- to_ulong
 - std::bitset< _Nb >, [1775](#)
 - std::tr2::dynamic_bitset< _WordT, _Alloc >, [2018](#)
- Todo List, [2](#)
- tolower
 - std, [709](#)
 - std::__ctype_abstract_base< _CharT >, [834](#), [835](#)
 - std::ctype< _CharT >, [1895](#)
 - std::ctype< char >, [1907](#)
 - std::ctype< wchar_t >, [1920](#), [1921](#)
 - std::ctype_byname< _CharT >, [1934](#)
 - std::ctype_byname< char >, [1945](#)
- top
 - std::priority_queue< _Tp, _Sequence, _Compare >, [2644](#)
 - std::stack< _Tp, _Sequence >, [2803](#)
- toupper
 - std, [709](#)
 - std::__ctype_abstract_base< _CharT >, [835](#), [836](#)
 - std::ctype< _CharT >, [1896](#)
 - std::ctype< char >, [1908](#)
 - std::ctype< wchar_t >, [1921](#)
 - std::ctype_byname< _CharT >, [1935](#)
 - std::ctype_byname< char >, [1946](#)
- TR1 Mathematical Special Functions, [261](#)
 - assoc_laguerre, [263](#)
 - assoc_legendre, [263](#)
 - beta, [263](#)
 - comp_ellint_1, [264](#)
 - comp_ellint_2, [264](#)
 - comp_ellint_3, [264](#)
 - conf_hyperg, [264](#)
 - cyl_bessel_i, [264](#)
 - cyl_bessel_j, [264](#)
 - cyl_bessel_k, [264](#)
 - cyl_neumann, [265](#)
 - ellint_1, [265](#)
 - ellint_2, [265](#)
 - ellint_3, [265](#)
 - expint, [265](#)
 - hermite, [265](#)
 - hyperg, [265](#)
 - laguerre, [266](#)
 - legendre, [266](#)
 - riemann_zeta, [266](#)
 - sph_bessel, [266](#)
 - sph_legendre, [266](#)
 - sph_neumann, [266](#)
- trace_fn_imps.hpp, [3419](#), [3420](#)
- Traits, [153](#)
- traits.hpp, [3405](#)–[3407](#)
- traits_type
 - std::basic_ios< _CharT, _Traits >, [1290](#)
 - std::basic_istream< _CharT, _Traits >::sentry, [2738](#)
 - std::basic_streambuf< _CharT, _Traits >, [1543](#)
 - std::istreambuf_iterator< _CharT, _Traits >, [2254](#)
 - std::ostream_iterator< _Tp, _CharT, _Traits >, [2569](#)
 - std::ostreambuf_iterator< _CharT, _Traits >, [2573](#)
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, [3128](#)
- transform
 - Mutating, [39](#)
 - std::collate< _CharT >, [1843](#)

- std::collate_byname< _CharT >, [1848](#)
- std::regex_traits< _Ch_type >, [2701](#)
- transform_exclusive_scan
 - Generalized Numeric operations, [16](#)
- transform_inclusive_scan
 - Generalized Numeric operations, [17](#), [18](#)
- transform_minimal_n
 - __gnu_parallel::Settings, [1079](#)
- transform_primary
 - std::regex_traits< _Ch_type >, [2702](#)
- transform_reduce
 - Generalized Numeric operations, [18](#), [19](#)
- translate
 - std::regex_traits< _Ch_type >, [2702](#)
- translate_nocase
 - std::regex_traits< _Ch_type >, [2703](#)
- traps
 - std::__numeric_limits_base, [866](#)
 - std::numeric_limits< _Tp >, [2533](#)
- tree
 - __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >, [2914](#)
- tree_policy.hpp, [3446](#)
- tree_trace_base.hpp, [3441](#)
- trie
 - __gnu_pbds::trie< Key, Mapped, ATraits, Tag, Node_Update, _Alloc >, [2926](#), [2927](#)
- trie_policy.hpp, [3447](#)
- trie_policy_base.hpp, [3442](#)
- trie_string_access_traits_imp.hpp, [3442](#)
- trivial_iterator_difference_type
 - Tags, [152](#)
- true_type
 - Metaprogramming, [399](#)
- trunc
 - std::num_punct< _CharT >, [2556](#)
 - std::num_punct_byname< _CharT >, [2561](#)
- trunc
 - std::basic_fstream< _CharT, _Traits >, [1239](#)
 - std::basic_ifstream< _CharT, _Traits >, [1283](#)
 - std::basic_ios< _CharT, _Traits >, [1306](#)
 - std::basic_iostream< _CharT, _Traits >, [1352](#)
 - std::basic_istream< _CharT, _Traits >, [1389](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1431](#)
 - std::basic_ofstream< _CharT, _Traits >, [1464](#)
 - std::basic_ostream< _CharT, _Traits >, [1496](#)
 - std::basic_ostreamstream< _CharT, _Traits, _Alloc >, [1529](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1725](#)
 - std::ios_base, [2201](#)
- try_emplace
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2361](#), [2362](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [3007](#), [3008](#)
- try_lock
 - Mutexes, [124](#)
- try_to_lock
 - Mutexes, [125](#)
- tuple, [3554](#), [3555](#)
- tuple_cat
 - Utilities, [350](#)
- type
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >, [1870](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >, [1870](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >, [1871](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI >, [1871](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >, [1872](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >, [1873](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >, [1873](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >, [1874](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >, [1874](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >, [1875](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >, [1875](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI >, [1876](#)
 - __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >, [1867](#)
 - __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >, [1868](#)
 - __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag,

- null_type >, 1868
 - __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type >, 1869
 - __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >, 1869
 - __gnu_pbds::detail::default_comb_hash_fn, 1954
 - __gnu_pbds::detail::default_eq_fn< Key >, 1957
 - __gnu_pbds::detail::default_hash_fn< Key >, 1957
 - __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >, 1959
 - __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >, 1959
 - __gnu_pbds::detail::default_trie_access_traits<std::basic_string< Char, Char_Traits, std::allocator< char > > >, 1960
 - __gnu_pbds::detail::default_update_policy, 1961
 - __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, true >, 2042
 - std::aligned_union< _Len, _Types >, 1090
 - std::any, 1114
 - std::experimental::fundamentals_v1::any, 1117
- Type-safe container of any type, 316
 - any_cast, 317–319
 - swap, 319
- type_identity_t
 - Metaprogramming, 399
- type_traits, 3557, 3561
- type_traits.h, 3457
- type_utils.hpp, 3442
- typeid, 3569
- typeid, 3154
- typelist.h, 3457
- types.h, 3529
- types_traits.hpp, 3443
- u16streampos
 - std, 634
- u16string
 - __gnu_debug, 457
 - Strings, 302
- u32streampos
 - std, 634
- u32string
 - __gnu_debug, 457
 - Strings, 302
- u8path
 - File System, 169
- uflow
 - __gnu_cxx::enc_filebuf< _CharT >, 2035
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2820
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2837
- std::basic_filebuf< _CharT, _Traits >, 1186
- std::basic_streambuf< _CharT, _Traits >, 1553
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1674
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3139
- uncaught_exception
 - Exceptions, 139
- uncaught_exceptions
 - Exceptions, 139
- undeclare_no_pointers
 - Pointer Safety and Garbage Collection, 388
- undeclare_reachable
 - Pointer Safety and Garbage Collection, 388
- underflow
 - __gnu_cxx::enc_filebuf< _CharT >, 2036
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2821
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2837
- std::basic_filebuf< _CharT, _Traits >, 1186
- std::basic_streambuf< _CharT, _Traits >, 1553
- std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1674
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3139
- underlying_type_t
 - Metaprogramming, 400
- unexpected
 - Exceptions, 139
- unexpected_handler
 - Exceptions, 136
- unset
 - std::basic_fstream< _CharT, _Traits >, 1233
 - std::basic_ifstream< _CharT, _Traits >, 1277
 - std::basic_iostream< _CharT, _Traits >, 1345
 - std::basic_istream< _CharT, _Traits >, 1383
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1425
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1718
- Uniform Distributions, 254
 - operator!=, 254
 - operator<=, 255
 - operator>=, 255, 256
- uniform_int_dist.h, 3313
- uniform_int_distribution
 - std::uniform_int_distribution< _IntType >, 2959
- uniform_real_distribution
 - std::uniform_real_distribution< _RealType >, 2961
- uninitialized_copy
 - Memory, 363
- uninitialized_copy_n
 - Memory, 363
 - SGI, 163
- uninitialized_default_construct
 - Memory, 363
- uninitialized_default_construct_n
 - Memory, 364

- uninitialized_fill
 - Memory, [364](#)
- uninitialized_fill_n
 - Memory, [365](#)
- uninitialized_move
 - Memory, [365](#)
- uninitialized_move_n
 - Memory, [365](#)
- uninitialized_value_construct
 - Memory, [366](#)
- uninitialized_value_construct_n
 - Memory, [366](#)
- unique
 - Mutating, [40, 41](#)
 - std::forward_list< _Tp, _Alloc >, [2090, 2091](#)
 - std::list< _Tp, _Alloc >, [2306](#)
 - std::shared_ptr< _Tp >, [2783](#)
- unique_copy
 - Mutating, [41, 42](#)
- unique_copy.h, [3530](#)
- unique_copy_minimal_n
 - __gnu_parallel:: Settings, [1079](#)
- unique_lock.h, [3313](#)
- unique_ptr
 - std::unique_ptr< _Tp, _Dp >, [2966, 2967](#)
 - std::unique_ptr< _Tp[], _Dp >, [2971, 2972](#)
- unique_ptr.h, [3314](#)
- unitbuf
 - std, [709](#)
 - std::basic_fstream< _CharT, _Traits >, [1239](#)
 - std::basic_ifstream< _CharT, _Traits >, [1283](#)
 - std::basic_ios< _CharT, _Traits >, [1306](#)
 - std::basic_iostream< _CharT, _Traits >, [1352](#)
 - std::basic_istream< _CharT, _Traits >, [1389](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1431](#)
 - std::basic_ofstream< _CharT, _Traits >, [1465](#)
 - std::basic_ostream< _CharT, _Traits >, [1497](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1529](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1725](#)
 - std::ios_base, [2201](#)
- Unordered Associative, [130](#)
- unordered_map, [3570, 3572](#)
 - std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, [2985–2987](#)
- unordered_map.h, [3315](#)
- unordered_multimap
 - std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3017–3019](#)
- unordered_multiset
 - std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3044–3046](#)
- unordered_set, [3573, 3575](#)
 - std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3072, 3073](#)
- unordered_set.h, [3317](#)
- unsetf
 - std::basic_fstream< _CharT, _Traits >, [1233](#)
 - std::basic_ifstream< _CharT, _Traits >, [1277](#)
 - std::basic_ios< _CharT, _Traits >, [1300](#)
 - std::basic_iostream< _CharT, _Traits >, [1346](#)
 - std::basic_istream< _CharT, _Traits >, [1384](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1425](#)
 - std::basic_ofstream< _CharT, _Traits >, [1458](#)
 - std::basic_ostream< _CharT, _Traits >, [1490](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1523](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1719](#)
 - std::ios_base, [2196](#)
- unshift
 - std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >, [820](#)
 - std::codecvt< _InternT, _ExternT, _StateT >, [1806](#)
 - std::codecvt< _InternT, _ExternT, encoding_state >, [1811](#)
 - std::codecvt< char, char, mbstate_t >, [1816](#)
 - std::codecvt< char16_t, char, mbstate_t >, [1821](#)
 - std::codecvt< char32_t, char, mbstate_t >, [1826](#)
 - std::codecvt< wchar_t, char, mbstate_t >, [1831](#)
 - std::codecvt_byname< _InternT, _ExternT, _StateT >, [1837](#)
- update_fn_imps.hpp, [3435](#)
- upper_bound
 - Binary Search, [90](#)
 - std::map< _Key, _Tp, _Compare, _Alloc >, [2362–2364](#)
 - std::multimap< _Key, _Tp, _Compare, _Alloc >, [2453, 2454](#)
 - std::multiset< _Key, _Compare, _Alloc >, [2478, 2480](#)
 - std::set< _Key, _Compare, _Alloc >, [2764, 2765](#)
- uppercase
 - std, [710](#)
 - std::basic_fstream< _CharT, _Traits >, [1240](#)
 - std::basic_ifstream< _CharT, _Traits >, [1283](#)
 - std::basic_ios< _CharT, _Traits >, [1306](#)
 - std::basic_iostream< _CharT, _Traits >, [1352](#)
 - std::basic_istream< _CharT, _Traits >, [1389](#)
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, [1431](#)
 - std::basic_ofstream< _CharT, _Traits >, [1465](#)
 - std::basic_ostream< _CharT, _Traits >, [1497](#)
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, [1529](#)
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, [1725](#)

1725
 std::ios_base, 2202
 use_count
 std::shared_ptr< _Tp >, 2783
 use_facet
 Locales, 181
 std::locale, 2315
 std::locale::id, 2170
 uses_allocator_args.h, 3319
 Utilities, 333

__addressof, 341
 __invoke, 341
 addressof, 341
 any_cast, 341–343
 declval, 344
 forward, 344
 forward_as_tuple, 344
 get, 344, 345
 make_any, 345, 346
 make_pair, 346
 move, 346
 move_if_noexcept, 347
 nullopt, 350
 operator!=, 347
 operator<, 347
 operator<=, 348
 operator>, 348
 operator>=, 348
 operator==, 348
 pair, 340
 piecewise_construct, 350
 swap, 348, 349
 tie, 349
 tuple_cat, 350

utility, 3576
 utility.h, 3320

valarray, 3577
 Numeric Arrays, 229, 230
 std::valarray< _Tp >, 3096
 valarray_after.h, 3321
 valarray_array.h, 3331
 valarray_array.tcc, 3339
 valarray_before.h, 3339
 valid_prefix

__gnu_pbds::detail::pat_trie_base::Node_citer<
 Node, Leaf, Head, Inode, _Clterator, Iterator,
 _Alloc >, 1003
 __gnu_pbds::detail::pat_trie_base::Node_iter<
 Node, Leaf, Head, Inode, _Clterator, Iterator,
 _Alloc >, 1009

value
 std::error_code, 2049
 std::error_condition, 2050

std::regex_traits< _Ch_type >, 2703
 value_comp
 std::map< _Key, _Tp, _Compare, _Alloc >, 2364
 std::multimap< _Key, _Tp, _Compare, _Alloc >,
 2454
 std::multiset< _Key, _Compare, _Alloc >, 2480
 std::set< _Key, _Compare, _Alloc >, 2766
 value_compare
 std::set< _Key, _Compare, _Alloc >, 2748
 value_type
 __gnu_pbds::detail::bin_search_tree_const_node_it<
 Node, Const_Iterator, Iterator, _Alloc >, 1732
 __gnu_pbds::detail::bin_search_tree_node_it<
 Node, Const_Iterator, Iterator, _Alloc >, 1736
 __gnu_pbds::detail::binary_heap_const_iterator<
 Value_Type, Entry, Simple, _Alloc >, 1745
 __gnu_pbds::detail::binary_heap_point_const_iterator<
 Value_Type, Entry, Simple, _Alloc >, 1749
 __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator<
 Node, _Alloc >, 2266
 __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator<
 Node, _Alloc >, 2270
 std::allocator_traits< _Alloc >, 1095
 std::allocator_traits< allocator< _Tp > >, 1100
 std::allocator_traits< allocator< void > >, 1104
 std::allocator_traits< pmr::polymorphic_allocator<
 _Tp > >, 1108
 std::back_insert_iterator< _Container >, 1157
 std::complex< _Tp >, 1853
 std::front_insert_iterator< _Container >, 2096
 std::insert_iterator< _Container >, 2183
 std::istream_iterator< _Tp, _CharT, _Traits, _Dist >,
 2251
 std::istreambuf_iterator< _CharT, _Traits >, 2254
 std::iterator< _Category, _Tp, _Distance, _Pointer,
 _Reference >, 2260
 std::ostream_iterator< _Tp, _CharT, _Traits >, 2570
 std::ostreambuf_iterator< _CharT, _Traits >, 2573
 std::raw_storage_iterator< _OutputIterator, _Tp >,
 2671
 std::set< _Key, _Compare, _Alloc >, 2748
 std::unordered_map< _Key, _Tp, _Hash, _Pred, _Al-
 loc >, 2985
 std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
 _Alloc >, 3017
 std::unordered_multiset< _Value, _Hash, _Pred, _Al-
 loc >, 3044
 std::unordered_set< _Value, _Hash, _Pred, _Alloc
 >, 3071
 Variable template for type traits, 319
 Variable templates for type traits, 402
 is_bounded_array_v, 405
 is_layout_compatible_v, 405
 is_pointer_interconvertible_base_of_v, 405

- is_unbounded_array_v, 406
- variant, 3577
- vector, 3581, 3582
 - std::__debug::vector< _Tp, _Allocator >, 3102
 - std::vector< _Tp, _Alloc >, 3105–3107
- vector.tcc, 3340
- void_pointer
 - __gnu_cxx::__alloc_traits< _Alloc, typename >, 806
 - std::allocator_traits< _Alloc >, 1095
 - std::allocator_traits< allocator< _Tp > >, 1100
 - std::allocator_traits< allocator< void > >, 1104
 - std::allocator_traits< pmr::polymorphic_allocator< _Tp > >, 1108
- void_t
 - Detection idiom, 314
 - Metaprogramming, 400
- vstring.h, 3458
- vstring.tcc, 3461
- vstring_fwd.h, 3462
- vstring_util.h, 3462
- wbuffer_convert
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3128
- wcerr
 - std, 711
- wcin
 - std, 711
- wclog
 - std, 711
- wcout
 - std, 711
- wregex_token_iterator
 - Regular Expressions, 272
- wcsub_match
 - Regular Expressions, 272
- weak_from_this
 - std::enable_shared_from_this< _Tp >, 2020
- weak_type
 - std::shared_ptr< _Tp >, 2776
- weeks
 - Time, 413
- wfilebuf
 - I/O, 172
- wfstream
 - I/O, 172
- what
 - __gnu_cxx::forced_error, 2070
 - __gnu_cxx::recursive_init_error, 2686
 - __gnu_pbds::container_error, 1877
 - __gnu_pbds::insert_error, 2181
 - __gnu_pbds::join_error, 2261
 - __gnu_pbds::resize_error, 2706
 - std::bad_alloc, 1159
 - std::bad_any_cast, 1160
 - std::bad_cast, 1161
 - std::bad_exception, 1162
 - std::bad_function_call, 1163
 - std::bad_optional_access, 1164
 - std::bad_typeid, 1165
 - std::bad_weak_ptr, 1166
 - std::domain_error, 2002
 - std::exception, 2053
 - std::experimental::filesystem::v1::filesystem_error, 2065
 - std::experimental::fundamentals_v1::bad_any_cast, 1160
 - std::experimental::fundamentals_v1::bad_optional_access, 1165
 - std::filesystem::filesystem_error, 2066
 - std::future_error, 2108
 - std::invalid_argument, 2187
 - std::ios_base::failure, 2063
 - std::length_error, 2272
 - std::logic_error, 2318
 - std::out_of_range, 2575
 - std::overflow_error, 2584
 - std::range_error, 2657
 - std::regex_error, 2691
 - std::runtime_error, 2718
 - std::system_error, 2867
 - std::underflow_error, 2957
- widen
 - std::__ctype_abstract_base< _CharT >, 836
 - std::basic_fstream< _CharT, _Traits >, 1233
 - std::basic_ifstream< _CharT, _Traits >, 1277
 - std::basic_ios< _CharT, _Traits >, 1301
 - std::basic_iostream< _CharT, _Traits >, 1346
 - std::basic_istream< _CharT, _Traits >, 1384
 - std::basic_istreamstream< _CharT, _Traits, _Alloc >, 1425
 - std::basic_ofstream< _CharT, _Traits >, 1459
 - std::basic_ostream< _CharT, _Traits >, 1491
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1523
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1719
 - std::ctype< _CharT >, 1897
 - std::ctype< char >, 1908, 1909
 - std::ctype< wchar_t >, 1922
 - std::ctype_byname< _CharT >, 1936
 - std::ctype_byname< char >, 1947
- width
 - std::basic_fstream< _CharT, _Traits >, 1234
 - std::basic_ifstream< _CharT, _Traits >, 1278
 - std::basic_ios< _CharT, _Traits >, 1301
 - std::basic_iostream< _CharT, _Traits >, 1346
 - std::basic_istream< _CharT, _Traits >, 1384

- std::basic_istream< _CharT, _Traits, _Alloc >, 1426
- std::basic_ofstream< _CharT, _Traits >, 1459
- std::basic_ostream< _CharT, _Traits >, 1491
- std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1523, 1524
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 1719
- std::ios_base, 2197
- wifstream
 - I/O, 172
- wios
 - I/O, 172
- wiostream
 - I/O, 172
- wistream
 - I/O, 172
- wistringstream
 - I/O, 172
- wofstream
 - I/O, 172
- workstealing.h, 3530
- wostream
 - I/O, 172
- wostringstream
 - I/O, 172
- wregex
 - Regular Expressions, 272
- write
 - std::basic_fstream< _CharT, _Traits >, 1234
 - std::basic_istream< _CharT, _Traits >, 1347
 - std::basic_ofstream< _CharT, _Traits >, 1460
 - std::basic_ostream< _CharT, _Traits >, 1492
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1524
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1720
- ws
 - std, 710
- wsregex_token_iterator
 - Regular Expressions, 272
- wssub_match
 - Regular Expressions, 273
- wstreambuf
 - I/O, 173
- wstreampos
 - std, 634
- wstring
 - Strings, 302
- wstring_convert
 - std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >, 3146
- wstringbuf
 - I/O, 173
- wstringstream
 - I/O, 173
- xalloc
 - std::basic_fstream< _CharT, _Traits >, 1235
 - std::basic_ifstream< _CharT, _Traits >, 1278
 - std::basic_ios< _CharT, _Traits >, 1302
 - std::basic_iostream< _CharT, _Traits >, 1347
 - std::basic_istream< _CharT, _Traits >, 1385
 - std::basic_istream< _CharT, _Traits, _Alloc >, 1426
 - std::basic_ofstream< _CharT, _Traits >, 1460
 - std::basic_ostream< _CharT, _Traits >, 1492
 - std::basic_ostringstream< _CharT, _Traits, _Alloc >, 1524
 - std::basic_stringstream< _CharT, _Traits, _Alloc >, 1720
 - std::ios_base, 2197
- xsgn
 - __gnu_cxx::enc_filebuf< _CharT >, 2036
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2821
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2838
 - std::basic_filebuf< _CharT, _Traits >, 1187
 - std::basic_streambuf< _CharT, _Traits >, 1554
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1675
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3139
- xsgn
 - __gnu_cxx::enc_filebuf< _CharT >, 2037
 - __gnu_cxx::stdio_filebuf< _CharT, _Traits >, 2822
 - __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 2838
 - std::basic_filebuf< _CharT, _Traits >, 1187
 - std::basic_streambuf< _CharT, _Traits >, 1554
 - std::basic_stringbuf< _CharT, _Traits, _Alloc >, 1675
 - std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3140
- years
 - Time, 413
- yield
 - std::this_thread, 786