**Containers**

**1.<array>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11) and [TR1](https://en.wikipedia.org/wiki/C%2B%2B_Technical_Report_1). Provides the container class template [std::array](https://en.wikipedia.org/wiki/Array_(C%2B%2B)), a container for a fixed sized array.

**2.<bitset>**

Provides the specialized container class [std::bitset](https://en.wikipedia.org/w/index.php?title=Bitset_(C%2B%2B)&action=edit&redlink=1), a [bit array](https://en.wikipedia.org/wiki/Bit_array).

**3.<deque>**

Provides the container class template [std::deque](https://en.wikipedia.org/wiki/Deque_(C%2B%2B)), a [double-ended queue](https://en.wikipedia.org/wiki/Deque).

**4.<forward\_list>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11) and [TR1](https://en.wikipedia.org/wiki/C%2B%2B_Technical_Report_1). Provides the container class template [std::forward\_list](https://en.wikipedia.org/wiki/Forward_list_(C%2B%2B)), a [singly linked list](https://en.wikipedia.org/wiki/Linked_list).

**5.<list>**

Provides the container class template [std::list](https://en.wikipedia.org/wiki/List_(C%2B%2B)), a [doubly linked list](https://en.wikipedia.org/wiki/Doubly_linked_list).

**6.<map>**

Provides the container class templates [std::map](https://en.wikipedia.org/wiki/Map_(C%2B%2B)) and std::multimap, sorted [associative array](https://en.wikipedia.org/wiki/Associative_array) and [multimap](https://en.wikipedia.org/wiki/Multimap_(data_structure)).

**7.<queue>**

Provides the container adapter class [std::queue](https://en.wikipedia.org/w/index.php?title=Queue_(C%2B%2B)&action=edit&redlink=1), a single-ended [queue](https://en.wikipedia.org/wiki/Queue_(data_structure)), and std::priority\_queue, a [priority queue](https://en.wikipedia.org/wiki/Priority_queue).

**8.<set>**

Provides the container class templates [std::set](https://en.wikipedia.org/wiki/Set_(C%2B%2B)) and std::multiset, sorted associative containers or [sets](https://en.wikipedia.org/wiki/Set_(computer_science)).

**9.<stack>**

Provides the container adapter class [std::stack](https://en.wikipedia.org/wiki/Stack_(C%2B%2B)), a [stack](https://en.wikipedia.org/wiki/Stack_(data_structure)).

**10.<unordered\_map>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11) and [TR1](https://en.wikipedia.org/wiki/C%2B%2B_Technical_Report_1). Provides the container class template [std::unordered\_map](https://en.wikipedia.org/wiki/Unordered_map_(C%2B%2B)) and std::unordered\_multimap, [hash tables](https://en.wikipedia.org/wiki/Hash_table).

**11.<unordered\_set>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11) and [TR1](https://en.wikipedia.org/wiki/C%2B%2B_Technical_Report_1). Provides the container class template [std::unordered\_set](https://en.wikipedia.org/wiki/Unordered_set_(C%2B%2B)) and std::unordered\_multiset.

**12.<vector>**

Provides the container class template [std::vector](https://en.wikipedia.org/wiki/Vector_(C%2B%2B)), a [dynamic array](https://en.wikipedia.org/wiki/Dynamic_array).

### General

**1.<algorithm>**

Provides definitions of many container [algorithms](https://en.wikipedia.org/wiki/Algorithm).

**2.<chrono>**

Provides time elements, such as std::chrono::duration, std::chrono::time\_point, and clocks.

**3.<functional>**

Provides several [function objects](https://en.wikipedia.org/wiki/Function_object), designed for use with the standard algorithms.

**4.<iterator>**

Provides classes and templates for working with [iterators](https://en.wikipedia.org/wiki/Iterator).

**5.<memory>**

Provides facilities for [memory management](https://en.wikipedia.org/wiki/Memory_management) in C++, including the class template std::unique\_ptr.

**6.<stdexcept>**

Contains standard exception classes such as std::logic\_error and std::runtime\_error, both derived from std::exception.

**7.<tuple>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11) and TR1. Provides a class template [std::tuple](https://en.wikipedia.org/w/index.php?title=Tuple_(C%2B%2B)&action=edit&redlink=1), a [tuple](https://en.wikipedia.org/wiki/Tuple).

**8.<utility>**

Provides the template class std::pair, for working with object pairs (two-member [tuples](https://en.wikipedia.org/wiki/Tuple)), and the namespace std::rel\_ops, for easier operator overloading.

**Localization**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=4)]

**1.<locale>**

Defines classes and declares functions that encapsulate and manipulate the information peculiar to a [locale](https://en.wikipedia.org/wiki/Locale_(computer_software)).

**2.<codecvt>**

Provides code conversion facets for various character encodings.

**Strings**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=5)]

**1.<string>**

Provides the C++ standard [string](https://en.wikipedia.org/wiki/String_(C%2B%2B)) classes and templates.

**2.<regex>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11). Provides utilities for pattern matching strings using [regular expressions](https://en.wikipedia.org/wiki/Regular_expressions).

**Streams and input/output**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=6)]

**1.<fstream>**

Provides facilities for file-based input and output. See [fstream](https://en.wikipedia.org/wiki/Fstream" \o "Fstream).

**2.<iomanip>**

Provides facilities to manipulate output formatting, such as the [base](https://en.wikipedia.org/wiki/Radix) used when formatting integers and the [precision](https://en.wikipedia.org/wiki/Arithmetic_precision) of [floating point](https://en.wikipedia.org/wiki/Floating_point) values.

**3.<ios>**

Provides several types and functions basic to the operation of iostreams.

**4.<iosfwd>**

Provides [forward declarations](https://en.wikipedia.org/wiki/Forward_declaration) of several I/O-related class templates.

**5.<iostream>**

Provides C++ input and output fundamentals. See [iostream](https://en.wikipedia.org/wiki/Iostream).

**6.<istream>**

Provides the template class std::istream and other supporting classes for input.

**7.<ostream>**

Provides the template class std::ostream and other supporting classes for output.

**8.<sstream>**

Provides the template class std::stringstream and other supporting classes for string manipulation.

**9.<streambuf>**

Provides reading and writing functionality to/from certain types of character sequences, such as external files or strings.

**Language support**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=7)]

**1.<exception>**

Provides several types and functions related to [exception handling](https://en.wikipedia.org/wiki/Exception_handling), including std::exception, the base class of all exceptions thrown by the Standard Library.

**2.<limits>**

Provides the template class std::numeric\_limits, used for describing properties of fundamental numeric types.

**3.<new>**

Provides operators new and delete and other functions and types composing the fundamentals of C++ [memory management](https://en.wikipedia.org/wiki/Memory_management).

**4.<typeinfo>**

Provides facilities for working with C++ [run-time type information](https://en.wikipedia.org/wiki/Run-time_type_information).

**Thread support library**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=8)]

**1.<thread>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11). Provide class and namespace for working with threads.

**2.<mutex>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11). 30.4-1. This section provides mechanisms for mutual exclusion: mutexes, locks, and call once.

**3.<condition\_variable>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11). 30.5-1. Condition variables provide synchronization primitives used to block a thread until notified by some other thread that some condition is met or until a system time is reached.

**4.<future>**

New in [C++11](https://en.wikipedia.org/wiki/C%2B%2B11). 30.6.1-1. Describes components that a C++ program can use to retrieve in one thread the result (value or exception) from a function that has run in the same thread or another thread.

**Numerics library**[[edit](https://en.wikipedia.org/w/index.php?title=C%2B%2B_Standard_Library&action=edit&section=9)]

Components that C++ programs may use to perform seminumerical operations.

**1.<complex>**

The header <complex> defines a class template, and numerous functions for representing and manipulating complex numbers.

**2.<random>**

Facility for generating (pseudo-)random numbers

**3.<valarray>**

Defines five class templates (valarray, slice\_array, gslice\_array, mask\_array, and indirect\_array), two classes (slice and gslice),and a series of related function templates for representing and manipulating arrays of values.

**4.<numeric>**

Generalized numeric operations.