std::stack

Defined in header <stack>

```
template<
    class T,
    class Container = std::deque<T>
> class stack;
```

The `std::stack` class is a container adaptor that gives the programmer the functionality of a stack - specifically, a LIFO (last-in, first-out) data structure.

The class template acts as a wrapper to the underlying container - only a specific set of functions is provided. The stack pushes and pops the element from the back of the underlying container, known as the top of the stack.

Template parameters

T - The type of the stored elements. The behavior is undefined if T is not the same type as `Container::value_type`.

Container - The type of the underlying container to use to store the elements. The container must satisfy the requirements of *SequenceContainer*. Additionally, it must provide the following functions with the usual semantics:

- `back()`, e.g., `std::vector::back()`,
- `push_back()`, e.g., `std::deque::push_back()`,
- `pop_back()`, e.g., `std::list::pop_back()`.

The standard containers `std::vector` (including `std::vector<bool>`), `std::deque` and `std::list` satisfy these requirements. By default, if no container class is specified for a particular stack class instantiation, the standard container `std::deque` is used.

Member types

Member type	Definition
`container_type`	`Container`
`value_type`	`Container::value_type`
`size_type`	`Container::size_type`
`reference`	`Container::reference`
`const_reference`	`Container::const_reference`

Member objects

Member name	Definition
`Container` **c**	the underlying container (protected member object)

Member functions

(constructor)	constructs the `stack` (public member function)
(destructor)	destructs the `stack` (public member function)
operator=	assigns values to the container adaptor (public member function)

Element access

top	accesses the top element (public member function)

Capacity

empty	checks whether the container adaptor is empty (public member function)
size	returns the number of elements (public member function)

Modifiers

push	inserts element at the top

	(public member function)
push_range (C++23)	inserts a range of elements at the top (public member function)
emplace (C++11)	constructs element in-place at the top (public member function)
pop	removes the top element (public member function)
swap (C++11)	swaps the contents (public member function)

Non-member functions

operator== **operator!=** **operator<** **operator<=** **operator>** **operator>=** **operator<=>** (C++20)	lexicographically compares the values of two `stacks` (function template)
std::swap(`std::stack`) (C++11)	specializes the `std::swap` algorithm (function template)

Helper classes

std::uses_allocator<`std::stack`> (C++11)	specializes the `std::uses_allocator` type trait (class template specialization)

Deduction guides (since C++17)

Notes

Feature-test macro	Value	Std	Feature
`__cpp_lib_containers_ranges`	`202202L`	(C++23)	Ranges construction and insertion for containers

Example

This section is incomplete
Reason: no example

Defect reports

The following behavior-changing defect reports were applied retroactively to previously published C++ standards.

DR	Applied to	Behavior as published	Correct behavior
LWG 307 (https://cplusplus.github.io/LWG/issue307)	C++98	`Container` could not be `std::vector<bool>`	allowed

See also

vector	dynamic contiguous array (class template)
vector<`bool`>	space-efficient dynamic bitset (class template specialization)
deque	double-ended queue (class template)
list	doubly-linked list (class template)