# vector Class

The STL vector class is a template class of sequence containers that arrange elements of a given type in a linear arrangement and allow fast random access to any element. They should be the preferred container for a sequence when random-access performance is at a premium.

template <   
 class Type,   
 class Allocator = allocator<Type>   
>   
class vector

## Parameters

*Type*

The element data type to be stored in the vector

*Allocator*

The type that represents the stored allocator object that encapsulates details about the vector's allocation and deallocation of memory. This argument is optional and the default value is **allocator***<Type>.*

## Remarks

Vectors allow constant time insertions and deletions at the end of the sequence. Inserting or deleting elements in the middle of a vector requires linear time. The performance of the deque Class container is superior with respect to insertions and deletions at the beginning and end of a sequence. The list Class container is superior with respect to insertions and deletions at any location within a sequence.

Vector reallocation occurs when a member function must increase the sequence contained in the vector object beyond its current storage capacity. Other insertions and erasures may alter various storage addresses within the sequence. In all such cases, iterators or references that point at altered portions of the sequence become invalid. If no reallocation happens, only iterators and references before the insertion/deletion point remain valid.

The vector<bool> Class is a full specialization of the template class vector for elements of type bool with an allocator for the underlying type used by the specialization.

The vector<bool> reference Class is a nested class whose objects are able to provide references to elements (single bits) within a vector<bool> object.

## Members

### Constructors

|  |  |
| --- | --- |
| vector | Constructs a vector of a specific size or with elements of a specific value or with a specific **allocator** or as a copy of some other vector. |

### Typedefs

|  |  |
| --- | --- |
| allocator\_type | A type that represents the **allocator** class for the vector object. |
| const\_iterator | A type that provides a random-access iterator that can read a const element in a vector. |
| const\_pointer | A type that provides a pointer to a const element in a vector. |
| const\_reference | A type that provides a reference to a const element stored in a vector for reading and performing const operations. |
| const\_reverse\_iterator | A type that provides a random-access iterator that can read any const element in the vector. |
| difference\_type | A type that provides the difference between the addresses of two elements in a vector. |
| iterator | A type that provides a random-access iterator that can read or modify any element in a vector. |
| pointer | A type that provides a pointer to an element in a vector. |
| reference | A type that provides a reference to an element stored in a vector. |
| reverse\_iterator | A type that provides a random-access iterator that can read or modify any element in a reversed vector. |
| size\_type | A type that counts the number of elements in a vector. |
| value\_type | A type that represents the data type stored in a vector. |

### Member Functions

|  |  |
| --- | --- |
| assign | Erases a vector and copies the specified elements to the empty vector. |
| at | Returns a reference to the element at a specified location in the vector. |
| back | Returns a reference to the last element of the vector. |
| begin | Returns a random-access iterator to the first element in the vector. |
| capacity | Returns the number of elements that the vector could contain without allocating more storage. |
| cbegin | Returns a random-access const iterator to the first element in the vector. |
| cend | Returns a random-access const iterator that points just beyond the end of the vector. |
| crbegin | Returns a const iterator to the first element in a reversed vector. |
| crend | Returns a const iterator to the end of a reversed vector. |
| clear | Erases the elements of the vector. |
| data | Returns a pointer to the first element in the vector. |
| emplace | Inserts an element constructed in place into the vector at a specified position. |
| emplace\_back | Adds an element constructed in place to the end of the vector. |
| empty | Tests if the vector container is empty. |
| end | Returns a random-access iterator that points to the end of the vector. |
| erase | Removes an element or a range of elements in a vector from specified positions. |
| front | Returns a reference to the first element in a vector. |
| get\_allocator | Returns an object to the **allocator** class used by a vector. |
| insert | Inserts an element or a number of elements into the vector at a specified position. |
| max\_size | Returns the maximum length of the vector. |
| pop\_back | Deletes the element at the end of the vector. |
| push\_back | Add an element to the end of the vector. |
| rbegin | Returns an iterator to the first element in a reversed vector. |
| rend | Returns an iterator to the end of a reversed vector. |
| reserve | Reserves a minimum length of storage for a vector object. |
| resize | Specifies a new size for a vector. |
| shrink\_to\_fit | Discards excess capacity. |
| size | Returns the number of elements in the vector. |
| swap | Exchanges the elements of two vectors. |

### Operators

|  |  |
| --- | --- |
| operator[] | Returns a reference to the vector element at a specified position. |
| operator= | Replaces the elements of the vector with a copy of another vector. |

## Requirements

**Header:** <vector>

Namespace: std