CENG 218 Data Structures

Sequence Containers (array, vector, list)

These are compiled from <https://cplusplus.com/reference/vector/vector/>

C++ vectors will not be used with all their features and methods to better discuss and understand the topic.

**Our class will use and discuss them with the following limited features.**

Vectors are sequence containers representing arrays that can change in size.
Just like arrays, vectors use contiguous storage locations for their elements, which means that their elements can also be accessed using offsets on regular pointers to its elements, and just as efficiently as in arrays. But unlike arrays, their size can change dynamically, with their storage being handled automatically by the container.
Internally, vectors use a dynamically allocated array to store their elements. This array may need to be reallocated in order to grow in size when new elements are inserted, which implies allocating a new array and moving all elements to it. This is a relatively expensive task in terms of processing time, and thus, vectors do not reallocate each time an element is added to the container.

Instead, vector containers may allocate some extra storage to accommodate for possible growth, and thus the container may have an actual [capacity](https://cplusplus.com/vector%3A%3Acapacity) greater than the storage strictly needed to contain its elements (i.e., its [size](https://cplusplus.com/vector%3A%3Asize)). Libraries can implement different strategies for growth to balance between memory usage and reallocations, but in any case, reallocations should only happen at logarithmically growing intervals of [size](https://cplusplus.com/vector%3A%3Asize) so that the insertion of individual elements at the end of the vector can be provided with *amortized constant time* complexity (see [push\_back](https://cplusplus.com/vector%3A%3Apush_back)).
Therefore, compared to arrays, vectors consume more memory in exchange for the ability to manage storage and grow dynamically in an efficient way.
Compared to the other dynamic sequence containers ([deques](https://cplusplus.com/deque), [lists](https://cplusplus.com/list) and [forward\_lists](https://cplusplus.com/forward_list)), vectors are very efficient accessing its elements (just like arrays) and relatively efficient adding or removing elements from its [end](https://cplusplus.com/vector%3A%3Aend). For operations that involve inserting or removing elements at positions other than the end, they perform worse than the others, and have less consistent iterators and references than [lists](https://cplusplus.com/list) and [forward\_lists](https://cplusplus.com/forward_list).

**Container properties**

* Sequence :Elements in sequence containers are ordered in a strict linear sequence. Individual elements are accessed by their position in this sequence.
* Dynamic array :Allows direct access to any element in the sequence, even through pointer arithmetics, and provides relatively fast addition/removal of elements at the end of the sequence.

**Member functions**

[**(constructor)**](https://cplusplus.com/reference/vector/vector/vector/) : Construct vector (public member function)

* Empty container constructor (default constructor): Constructs an [empty](https://cplusplus.com/vector%3A%3Aempty) container, with no elements.
* Fill constructor :Constructs a container with *n* elements. Each element is a copy of *val* (if provided).
* Copy constructor (and copying with allocator) :Constructs a container with a copy of each of the elements in *x*, in the same order.

[**(destructor)**](https://cplusplus.com/reference/vector/vector/~vector/) : Vector destructor (public member function)

[**operator=**](https://cplusplus.com/reference/vector/vector/operator%3D/) : Assign content (public member function)

Parameters:

* n : Initial container size (i.e., the number of elements in the container at construction).
Member type size\_type is an unsigned integral type.
* val :Value to fill the container with. Each of the *n* elements in the container will be initialized to a copy of this value.
Member type value\_type is the type of the elements in the container, defined in [vector](https://cplusplus.com/vector) as an alias of its first template parameter (T).

### Example

*#include <iostream>*

*#include <vector>*

*int* main ()

{

 std::vector<*int*> first; *// empty vector of ints*

 std::vector<*int*> second (4,100); *// four ints with value 100*

 std::vector<*int*> fourth (third);

}

**Capacity**:

[**size**](https://cplusplus.com/reference/vector/vector/size/) : Return size (public member function)

[**max\_size**](https://cplusplus.com/reference/vector/vector/max_size/) : Return maximum size (public member function)

[**resize**](https://cplusplus.com/reference/vector/vector/resize/) : Change size (public member function)

[**capacity**](https://cplusplus.com/reference/vector/vector/capacity/) : Return size of allocated storage capacity (public member function)

[**empty**](https://cplusplus.com/reference/vector/vector/empty/) : Test whether vector is empty (public member function)

[**reserve**](https://cplusplus.com/reference/vector/vector/reserve/) : Request a change in capacity (public member function)

[**shrink\_to\_fit**](https://cplusplus.com/reference/vector/vector/shrink_to_fit/)  : Shrink to fit (public member function)

**Element access**:

[**operator[]**](https://cplusplus.com/reference/vector/vector/operator%5B%5D/) : Access element (public member function)

[**at**](https://cplusplus.com/reference/vector/vector/at/) : Access element (public member function)

[**front**](https://cplusplus.com/reference/vector/vector/front/) : Access first element (public member function)

[**back**](https://cplusplus.com/reference/vector/vector/back/) : Access last element (public member function)

[**data**](https://cplusplus.com/reference/vector/vector/data/)  : Access data (public member function)

**Modifiers**:

[**push\_back**](https://cplusplus.com/reference/vector/vector/push_back/) : Add element at the end (public member function)

[**pop\_back**](https://cplusplus.com/reference/vector/vector/pop_back/) : Delete last element (public member function)

[**insert**](https://cplusplus.com/reference/vector/vector/insert/) : Insert elements (public member function)

[**erase**](https://cplusplus.com/reference/vector/vector/erase/) : Erase elements (public member function)

[**clear**](https://cplusplus.com/reference/vector/vector/clear/) : Clear content (public member function)