Vectors in C++

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 greater than the storage strictly needed to contain its elements (i.e., its size).

Therefore, compared to arrays, vectors consume more memory in exchange for the ability to manage storage and grow dynamically in an efficient way.

How to declare vectors?

First we have to include the vector header file
#include <vector>
Then for declaring,
vector<data_type> vector_name;
eg. vector<int> a;
For creating and initializing a vector

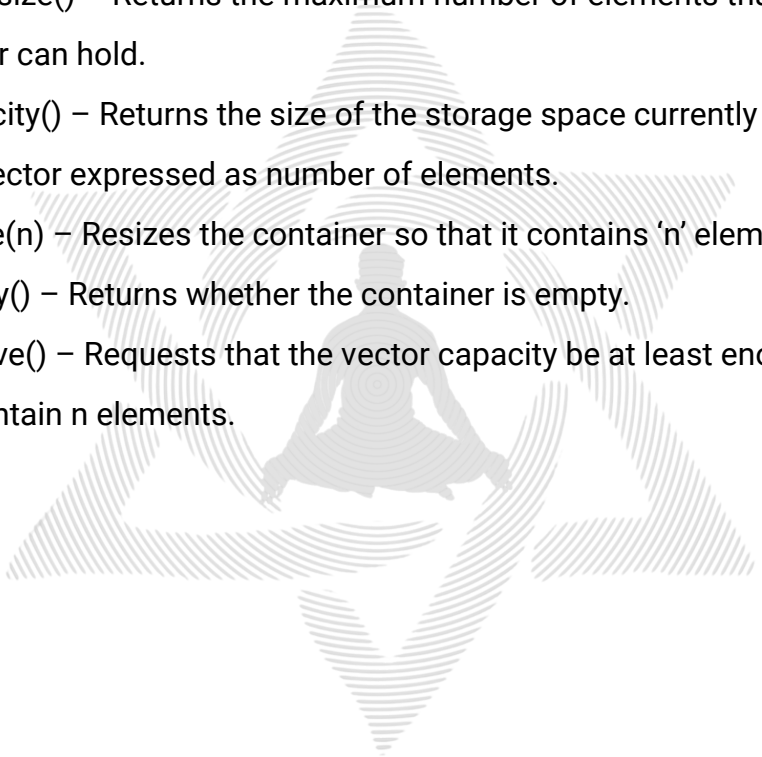
```
//You can create and initalise a vector
vector<int> a;
vector<int> b(5,10); //five int with value 10 - init a vector of zeros (n,0)
vector<int> c(b.begin(),b.end());
vector<int> d{1,2,3,10,14};
```

To access any specific element from a vector we can write(just like array)

vector_name[index]

Functions that we can use:

1. size() – Returns the number of elements in the vector.
2. max_size() – Returns the maximum number of elements that the vector can hold.
3. capacity() – Returns the size of the storage space currently allocated to the vector expressed as number of elements.
4. resize(n) – Resizes the container so that it contains ‘n’ elements.
5. empty() – Returns whether the container is empty.
6. reserve() – Requests that the vector capacity be at least enough to contain n elements.



For adding an element in vector:

1. assign() – It assigns new value to the vector elements by replacing old ones
2. push_back() – It push the elements into a vector from the back
3. pop_back() – It is used to pop or remove elements from a vector from the back.
4. insert() – It inserts new elements before the element at the specified position

We can also swap the contents of one vector with another vector of the same type(sizes may differ), by using the swap() function.

By using erase() function we can remove a particular element by specifying the position.
And by the clear() function, we can remove all the elements from the vector.