Date ____________ Page ________

Day - 37

Vectors in C++

* **Standard Template Library :**

⇒ It is a collection of predefined templates.

* **How to declare a vector :**

⇒ vector <data type> v;

⇒ vector <int> v;

* **How to insert value in vector :**

⇒ vector <int> v (size of vector, initialize);

⇒ vector < int > v (4);

⇒ vector < int > v (4, 2);

⇒
```
vector <int> a;
a. push_back (4);        [4]
a. push_back(8);         [4|8]
```

⇒ vector <int> v = { 2, 4, 6, 8, 10 }

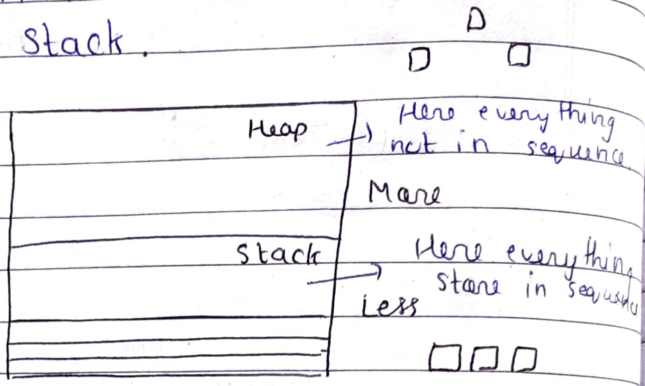
⇒
```
vector <int > v (5);
for (i=0; i<5; i++)
        cin>> v[i];
```

⇒ We take size of vector from the user, it is allow in vector.

⇒ And generally, this thing is not allow in array.

Date ________ Page ________

* Our RAM is divided in two parts - Heap & Stack.


Heap
Here every thing not in sequence
More
Stack
Here everything store in sequence
less

=) All the variables and our main function is store in stack that's why dynamic size array are not allow. ~~in~~

=) Because stack have less memory as compared to Heap.

=) But vector are store in heap.

=) Static memory allocation:
when we know the size at compile time.

=) Dynamic memory allocation
when we ~~kno~~ will get to know the size at runtime.

=) why vector creates double size of vector of its current size every time when it gets full?

=) Because it wants to insert elements in vector in O(1) time and make itself ready for any more insertion

Date   Page

=) v. push_back(2) $\boxed{2}$

=) v. push back(3) $\boxed{2 \mid \ }$ → $\boxed{2 \mid 3}$

* Remove value from vector

=) v. pop_back(); ⟶ O(1)
(delete last element of vector)

=) v. clear(); ⟶ O(n)
(delete all elements of the vector)

=) v. erase (v. begin+2); → O(n)
(If you want to delete particular element from the vector).

* Size and Capacity:

=) $\boxed{2 \mid 4 \mid 6 \mid 8 \mid 9 \mid \ \mid \ \mid \ }$

Size : 5 (No. of elements in vector)

capacity : 8 (No. of elements that can come in the vector)

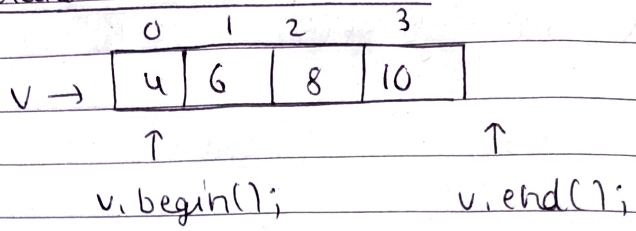
* Front → v. front (); ⟶ 2
(gives the first element)

* Back: v. back (); ⟶ 9
(gives the last element)

* Empty : v. empty (); →
1 → when empty
0 → when not empty

Date Page

* Iterator in a vector:



=> Print elements by using iterator

```
for(auto it= v.begin(); it != v.end();
 it++){
        cout << *it << " ";
 }
```

=> v.rbegin() → Reverse begin
=> v.rend( ) → Reverse end

* Sorting:
=> sort(v.begin(), v.end());
  (Increasing order)
=> sort(v.begin(), v.end(), greater<int>());
  (Decreasing order).