Vectors

Array like, have indices as well.
Dynamic in nature (difference b/w array & vectors)

STL (Std. Template Library)

- Stack
- Hash-Table
- Que

Have the implementations of the data structures.

- Vector
- Set

→ STL Container

Vectors

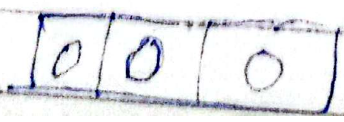
Can be resized dynamically
Syntax:

→ vector<int> vec_name

By default have 0 memory.
We have to include the header file as well "#include <vector>"

→ vector<int> = {1,2,3}
→ vector<int> = (3,0)

size of vector ← 3, 0 → value



for each loop

It is a special type of loop used in the vectors

for (int i :   vec_name)

It does not represents/gives the index. It gives the value directly

Iterator

Vector functions

→ size (gives the size of vector.
vec.size()

→ push_back (Adding going forward)
vec.push_back(25) [25|25|25] →

→ pop_back (deleting) (by-default the last one)
vec.pop_back()

→ front printing the first value
vec.front()

→ back
printing the last value vec.back().

→ at
Getting access at the particular index vec.at[0] etc.

Static VS Dynamic

①- Static
Memory gets allocated at the compile time.

int arr[5]

①- Dynamic
Memory gets allocated at the run-time.
Lets take a look on the vectors
In the start like
②— vector <int> = {1,2,3}
Memory has been allocated static
But if we apply the function
now, then the static memory gets
changes into dynamic and that's
why vectors are dynamic in nature
②— Arrays are static
③— Arrays use stack static memory
uses stacks
③— Vectors uses heaps, Dynamic memory
uses heap.

vector <int> vec_name

[0] gets double ← vec.push(0)
0

[0|1] ← vec.push(1)
gets double

[0|1|2|] ← vec.push(2)

At first 1 tries to fit in index 0, but it is filled so, a double capacited vector is created to store both of them and this goes on...

Vector

size of vector
No. of Elements

capacity
How many numbers can be stored