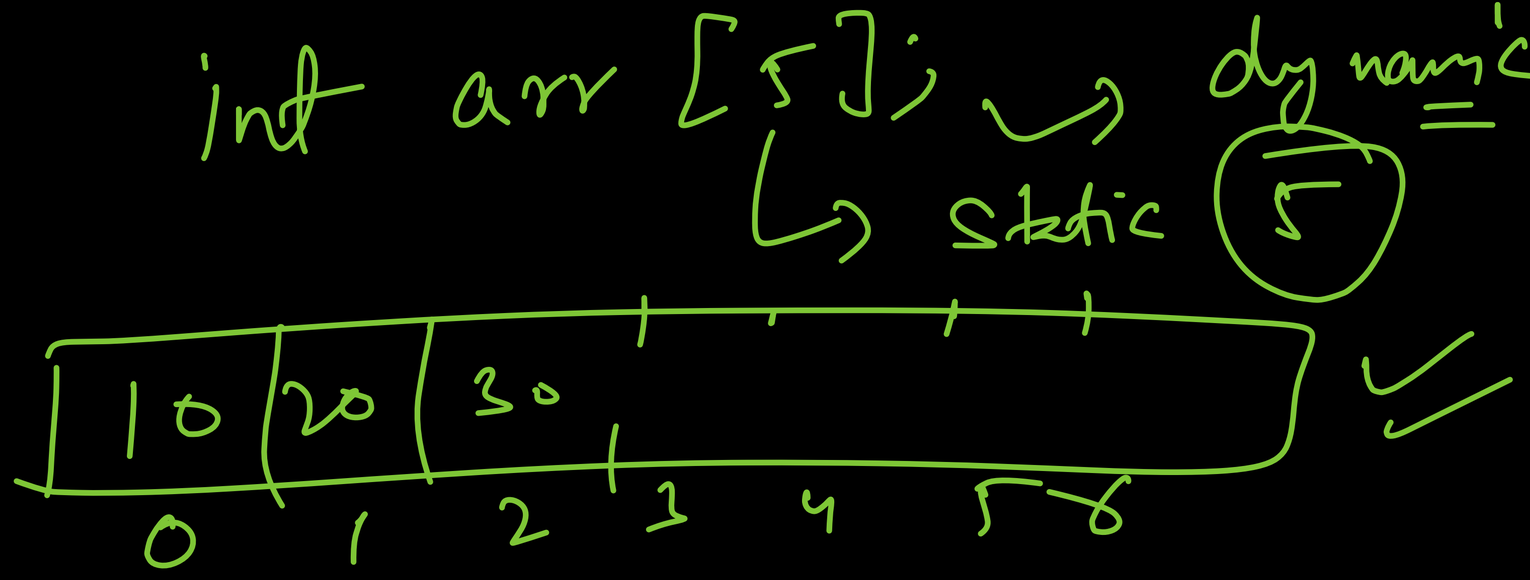
Vector STL in C++

CodeHelp Supreme Batch

Introduction

1. The Standard Template Library (STL) provides a collection of template classes and functions that offer common data structures and algorithms to make programming more efficient and convenient.
2. A vector in C++ is a dynamic array that can grow or shrink in size, making it a versatile and efficient data structure for storing and manipulating sequences of elements.



Features

1. **Contiguous Memory**: Elements in a vector are stored in contiguous memory locations, which makes it efficient for random access and iteration.

2. **Dynamic Sizing:** Unlike built-in arrays in C++, which have a fixed size, vector can dynamically resize itself as elements are added or removed. This dynamic sizing is managed internally, so you don't need to worry about memory management.

3. **Automatic Reallocation**: When a vector reaches its capacity and you try to add more elements, it automatically reallocates memory to accommodate the new elements. This allows you to work with dynamic-sized collections without worrying about memory management.

4. **Size and Capacity**: vector maintains two important properties: the size, which is the number of elements currently stored in the vector, and the capacity, which is the number of elements the vector can hold without reallocation.

5. **Array-Like Access**: You can access elements in a vector using array-like syntax, using square brackets ([]) or the at() member function.

⇒ int arr [5];

↳ Static arr → Static Mem. allocation

↳ dynamic array

⇒ user i/p 5

10	20	30	40	50
0	1	2	3	4

⇒ arr[5] = 80 // ???

int n;
cin >> n;

i/p → 50

int * arr = new int [n];

50 size

5th push Token

Dynamic memory allocation

⇒ Na to user se puchna hai

⇒ Just keep inserting the data.

⇒ Vector<int> v;

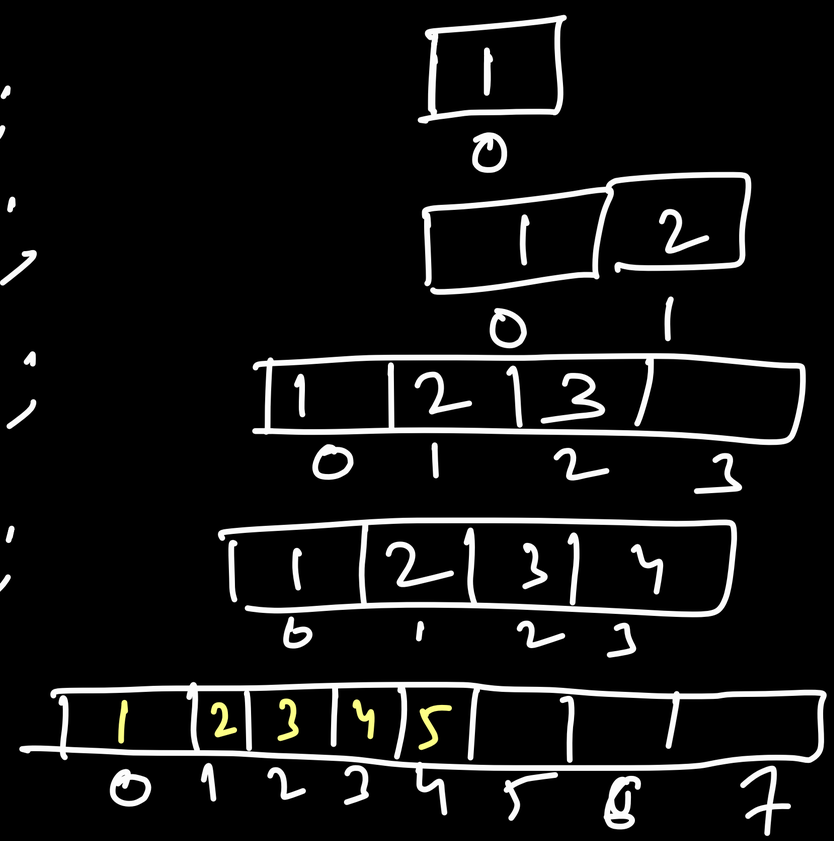
v.push_back(1);

" (2);

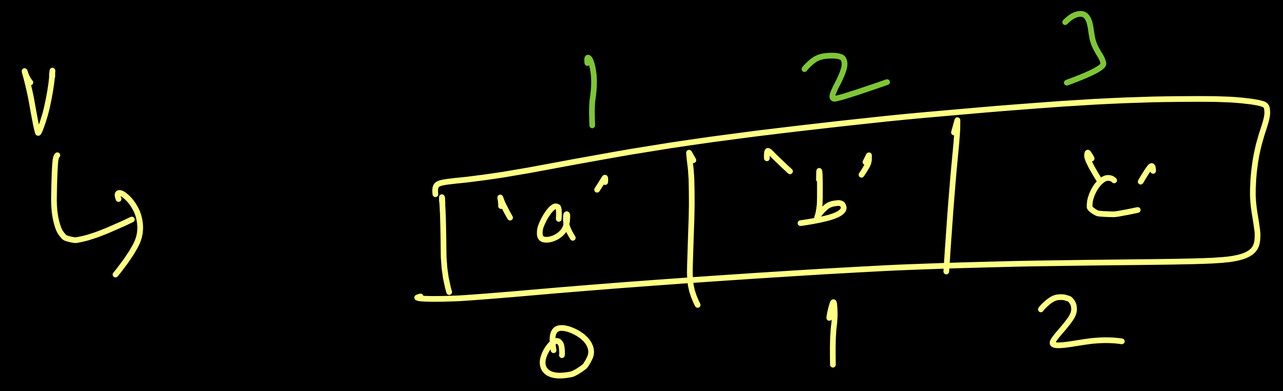
" (3);

" (4);

" (5);



Capacity ↓ Cap.	size
1	1
2	2
4	3
4	4
8	5



Front ⇒ v[0]

End ⇒ v[v.size() - 1]

v[3 - 1]

v[2] ⟶