**Vector Built-in Functions:**

1. **Constructor**

|  |  |  |
| --- | --- | --- |
| **Name** | **Details** | **Time Complexity** |
| **vector<type>v;** | Construct a vector with 0 elements. | O(1) |
| **vector<type>v(N);** | Construct a vector with N elements and the value will be garbage. | O(N) |
| **vector<type>v(N,V);** | Construct a vector with N elements and the value will be V. | O(N) |
| **vector<type>v(v2);** | Construct a vector by copying another vector v2. | O(N) |
| **vector<type>v(A,A+N);** | Construct a vector by copying all elements from an array A of size N.   int a[6] = {1, 2, 3, 4, 5, 6};   vector<int>v(a, a + 6); | O(N) |

1. **Capacity**

|  |  |  |
| --- | --- | --- |
| **Name** | **Details** | **Time Complexity** |
| **v.size()** | Returns the size of the vector. | O(1) |
| **v.max\_size()** | Returns the maximum size that the vector can hold. | O(1) |
| **v.capacity()** | Returns the current available capacity of the vector. | O(1) |
| **v.clear()** | Clears the vector elements. Do not delete the memory, only clear the value. | O(N) |
| **v.empty()** | Return true/false if the vector is empty or not. | O(1) |
| **v.resize()** | Change the size of the vector. | O(K); where K is the difference between new size and current size. |

1. **Modifiers**

|  |  |  |
| --- | --- | --- |
| **Name** | **Details** | **Time Complexity** |
| **v= or v.assign()** | Assign another vector. | O(N) if sizes are different, O(1) otherwise. |
| **v.push\_back()** | Add an element to the end. | O(1) |
| **v.pop\_back()** | Remove the last element. | O(1) |
| **v.insert()** | Insert elements at a specific position.  vector<int> v = {1, 2, 3};  v.insert(v.begin()+2, 100);  v.insert(v.begin()+2,v2.begin(), v2.end()); | O(N+K); where K is the number of elements to be inserted. |
| **v.erase()** | Delete elements from a specific position.  v.erase(v.begin()+3);  v.erase(v.begin()+1, v.begin()+4); | O(N+K); where K is the number of elements to be deleted. |
| **replace(v.begin(),v.end(),value,replace\_value)** | Replace all the value with replace\_value. Not under a vector.  replace(v.begin(), v.end()-1, 2, 100); | O(N) |
| **find(v.begin(),v.end(),V)** | Find the value V. Not under a vector.  1 way….  vector<int> v = {1, 2, 2, 4, 3, 5,1, 2,  4, 5, 3, 2}; vector<int>::iterator it;  it = find(v.begin(),v.end(),3);  cout << \*it<< endl;  2 way….  auto it= find(v.begin(),v.end(),3);  cout << \*it<< endl;  ‘  3 way….  auto it= find(v.begin(),v.end(),100);  if(it == v.end())    cout << "NOt Found";  else     cout << "Found"; | O(N) |

1. **Element access**

|  |  |  |
| --- | --- | --- |
| **Name** | **Details** | **Time Complexity** |
| **v[i]** | Access the ith element. | O(1) |
| **v.at(i)** | Access the ith element. | O(1) |
| **v.back()** | Access the last element. | O(1) |
| **v.front()** | Access the first element. | O(1) |

1. **Iterators**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Details** | **Time Complexity** | |
| **v.begin()** | Pointer to the first element. | O(1) | |
| **v.end()** | Pointer to the last element. | O(1) | |
|  |
| **sort()** | sort(nums.begin(),nums.end()); //assinding  sort(nums.begin(),nums.end(),greater<int>()); //des | |  |
| **max\_element()** | int i = \*max\_element(nums.begin(),nums.end()); | |  |
| **min\_element()** | \*min\_element(nums.begin(),nums.end()); | |  |
| **accumulate()** | int totalSum = accumulate(nums.begin(), nums.end(), 1); // Calculate total sum | |  |
| Duplicate remove -> | void ssort(vector<int>&v)  {      sort(v.begin(), v.end());      vector<int>::iterator ip;      ip = unique(v.begin(), v.end());      v.resize(distance(v.begin(), ip));  } | |  |