**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),, O(1) |
| **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());  *vector*<*int*> v;  v.insert(v.end(), 4, 100);*//v = {100,100,100,100}* | 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);  //remove all same value;  V{1,2,3,4,5,3,3}; I want to all 3 remove; output: {1,2,4,5}  v.erase(remove(v.begin(), v.end(),3), v.end()); | O(N+K); where K is the number of elements to be deleted. |
| **replace()** | Replace all the value with replace\_value. Not under a vector.  replace(v.begin(), v.end()-1, 2, 100);  (begin(),end(),value, replace\_value) | O(N) |
| **find()** | 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));  } |  |
| Vector রেন্ডমভাবে সাজানো  rand() | rand**()** হলো সি এবং সি++ প্রোগ্রামিং ভাষায় একটি ফাংশন যা রান্ডম ইন্টিজার তৈরি করে।  for (int i=0; i<5; i++) cout << rand() <<endl;  **//Vector randomly a**  vector<int> v={1,2,3,4,6,7,8};  int n = v.size();  int back = n;  for (int i = n - 1; i >= 0; i--)  {  int j = rand() % back;  swap(v[i], v[j]);  back--;  } output: {2 4 3 1 6 7 8}  **//Set seed using current time**  srand(time(0));  for (int i=0; i<5; i++) cout << rand() <<endl; | O(N) | |
| next\_permutation() | vector<int> g = {1, 2, 3, 4, 5, 6};      next\_permutation(g.begin(), g.end());      for (auto c:g) cout << c << " ";  vector<int> v = {1, 2, 3};  do  {  cout << v[0] << " " << v[1] << " " << v[2] << ln;  } while (next\_permutation(v.begin(), v.end())); | V = 1 2 3  Next permutation is  1 3 2  2 1 3  2 3 1  3 2 1  3 1 2 | |
| lower\_bound() and upper\_bound() | vec v = {1, 3, 4, 6, 8, 9};  auto i = lower\_bound(v.begin(),v.end(), 2);  auto j = upper\_bound(v.begin(),v.end(), 8);  cout << j-i << endl; // output: 4 | ২ থেকে ৮ এর মধ্যে কয়টা valu আছে তা বের করার জন্য, | |
| inner\_product() | vector<int> vec1 = {1, 2, 3};  vector<int> vec2 = {4, 5, 6};  int innerProd =  inner\_product(vec1.begin(), vec1.end(), vec2.begin(), 0);  int cnt = inner\_product(  students.begin(), students.end(), // প্রথম ভেক্টরের শুরু ও শেষ  seats.begin(), // দ্বিতীয় ভেক্টরের শুরু  0, // শুরু মান (cnt এর জন্য)  plus<int>(), // ফলাফলের জন্য বাইনারি অপারেশন (যোগ)  [](int a, int b) { // প্রতিটি উপাদান জোড়ার উপর প্রয়োগ করা ফাংশন  return abs(a - b); // কাস্টম ফাংশন, যা পার্থক্যের absolute value ফেরত দেয়  } ); | (1\*4)+(2\*5)+(3\*6) = 32 ans  Plus<int>() GLv‡b Gi cwie‡Z custom ফাংশন ব্যাবহার করা যাবে।  [](int a, int b) { return a + b; }, | |
| Partial\_sum() | vector<int> values = {1, 2, 3, 4, 5};  vector<int> result(values.size());  partial\_sum(values.begin(), values.end(), result.begin());  cout << "Partial sums:";  for (int x : result)  cout << " " << x;  //output : 1 3 6 10 15 |  | |
| Gcd()  Lcm() | int a = 12, b = 18;  int greatestCommonDivisor = gcd(a, b);  int leastCommonMultiple =  lcm(a, b); |  | |
| Shift target value to end; | shift the 0's to the end;  input = [1,0,4,0,2,0,0]  output = [1,4,2,0,0,0]  vector<int>v;  auto it = remove(begin(v),end(v),0);  fill(it, v.end(),0; |  | |
| Custom sort | Input: {-20, -6, -15, -1, 0, 10, 2, 8}  Output: {0, 2, 8, 10, -1, -6, -15, -20}  Bool custom(int a ,int b)  {  if(a<0 and b<0) return a>b;  if(a<0) return false;  if(b<0) return true;  return a<b;  } | Input: | |