**VECTORS IN C++**

Vectors uses array method to create a dynamic array(list);

Vectors are the same as dynamic arrays with the ability to resize itself automatically when an element is inserted or deleted, with their storage being handled automatically by the container.

**Initialization:- vector<datatype> variable\_name;**

**Header :- #include<vector>**

|  |  |
| --- | --- |
| **Function** | **Description** |
| **v.begin()** | Return a pointer to first element in the vector |
| **v.end()** | Returns a iterator to the last element of the vector |
| **v.rbegin()** | Returns a reverse iterator pointing to the last element in the vector (reverse beginning). It moves from last to first element |
| **v.rend();** | Returns a reverse iterator pointing to the theoretical element preceding the first element in the vector (considered as reverse end) |
| **v.cbegin()** | Returns a constant iterator pointing to the first element in the vector ie values cannot be changed |
| **v.cend()** | Returns a constant iterator pointing to the last element in the vector ie values cannot be changed |
| **v.crbegin()** | Returns a constant reverse iterator pointing to the last element in the vector (reverse beginning). It moves from last to first element.Values cannot be changed |
| **v.crend()** | Returns a constant reverse iterator pointing to the first element in the vector. It moves from last to first element.Values cannot be changed |
| **v.size()** | Return the size of the vector; |
| **v.max\_size()** | Return the max size the vector can hold(System limitations) |
| **v.capacity()** | Returns the size of the storage space currently allocated to the vector expressed as number of elements. (may or maynit be equal to size) |
| **v.resize(n)** | Resizes the container so that it contains ‘n’ elements. |
| **v.empty()** | If vector is empty or not |
| **v.shrink\_to\_fit()** | – Reduces the capacity of the container to fit its size and destroys all elements beyond the capacity. |
| **v.reserve(n)** | – Reduces the capacity of the container to fit its size and destroys all elements beyond the capacity. |
| **MODIFIERS:\_** | |
| **v.assign(x,y)** | Fills/replaces the vector with y or x times |
| **v.push\_back(n)** | Pushes the element n at the back of the vector |
| **v.pop\_back()** | Deletes element at the back; |
| **v.insert(x,y)** | Inserts **y** before **x** postion |
| **v.erase(x)** | Deletes at postion **x** |
| **v1.swap(v2);** | Swaps content of v1 and v2 |
| **v.emplace(x,y)** | It inserts y at position at X (can add multiple elements) |
| **v.emplace\_back(x,y)** | It inserts y at position at X (can add multiple elements) |
|  |  |
| **Accessing** | |
| **V[n]** | Returns a reference to the element at position ‘n’ in the vector |
| **v.at(g)** | Returns a reference to the element at position ‘g’ in the vector |
| **v.front()** | Returns a reference to the first element in the vector. |
| **v.back()** | Returns a reference to the flast element in the vector |
| **v.data()** | Return a pointer to the first element. Eg int \* ptr=g1.data;cout<<\*ptr; |