C++ STL Vector

August 31, 2019

[1]:
```
#include <iostream>
#include <vector>

using namespace std;
```

Array - static data structure, size to be defined while declaration. It doesn't increase it's size depending on requirement.

[2]:
```
template<class T>
void print(vector<T> arr) {
    for(int i=0; i<arr.size(); i++) {
        cout << arr[i];
    }
}
```

0.1 push_back()

[3]:
```
// Vector declaration
vector<int> v;
v.push_back(2);
v.push_back(0);
v.push_back(8);
print(v);
```

```
208
```

[4]:
```
vector<string> s;
s.push_back("Kushashwa ");
s.push_back("Ravi ");
s.push_back("Shrimali ");
print(s);
```

```
Kushashwa Ravi Shrimali
```

[5]:
```
// Size of a vector
int size = s.size();
cout << size;
```

```
3
```

0.2 pop_back()

[6]:
```
vector<string> sample;
sample.push_back("Kushashwa ");
sample.push_back("Ravi ");
sample.push_back("Shrimali ");
print(sample);
```

```
Kushashwa Ravi Shrimali
```

[7]:
```
cout << "Size before popping: " << sample.size() << endl;;
sample.pop_back();
cout << "Size after popping: " << sample.size() << endl;
```

```
Size before popping: 3
Size after popping: 2
```

[8]:
```
print(sample);
```

```
Kushashwa Ravi
```

0.3 Accessing Vector Elements

[9]:
```
print(sample);
```

```
Kushashwa Ravi
```

0.3.1 1. Indexing Numerically

[10]:
```
for(int i=0; i<sample.size(); i++) {
    cout << sample[i] << endl;
}
```

```
Kushashwa
Ravi
```

0.3.2 2. Using Iterator

[11]:
```
vector<int> int_v;
int_v.push_back(2);
int_v.push_back(0);
int_v.push_back(8);

vector<int>::iterator iter;
for(iter=int_v.begin(); iter!=int_v.end(); iter++) {
    cout << *iter;
```

```
}
```

208

0.4 Insert into Vector

[12]:
```
vector<int> vec;
vec.push_back(2);
vec.push_back(0);
vec.push_back(8);
print(vec);
```

208

[13]:
```
vec.insert(vec.begin()+1, 2);
```

[14]:
```
print(vec);
```

2208

[15]:
```
vec.insert(vec.end()-1, 4);
```

[16]:
```
print(vec);
```

22048

0.5 Erase from a Vector

Let's try to get the original vector (208) back. Note that, from the end, it starts from -1, -2, and so on. So when you do `vec.erase(vec.end() - 1)` - you're basically erasing the *last* element. But if you do `vec.erase(vec.begin()+1)` - you're erasing the *second* element from the beginning.

[17]:
```
print(vec);
```

22048

[18]:
```
vec.erase(vec.end()-2);
print(vec);
```

2208

[19]:
```
vec.erase(vec.begin()+1);
print(vec);
```

208

0.6 Clear Vector

[25]:
```
vec.clear();
print(vec);
cout << "Size: " << vec.size() << endl;
```

```
Size: 0
```

0.7 Sorting Vector

[26]:
```
vec.push_back(2);
vec.push_back(0);
vec.push_back(8);
```

[27]:
```
print(vec);
```

```
208
```

[28]:
```
cout << "Vector before sorting\n";
print(vec);
sort(vec.begin(), vec.end());
cout << "\nVector after sorting\n";
print(vec);
```

```
Vector before sorting
208
Vector after sorting
028
```