Abstraction and Paradigms of Programming

Bachelor of Engineering, Computer Engineering

II nd Year IV th Semester

Writing Iterators in Short

There are two ways to write iterators in short :

- Range based loops
- Auto keyword

Range based loops

For vectors

```
Main()
Vector<int>v={2,3,4,5,6};
For (int value : v){
Cout<<value<<endl;
}
```

For pair of vectors

```
Main()
Vector<pair<int,int>>v={{1,2},{2,3},{3,4}}
For(pair<int,int> value : v)
Cout<<value.first<<“ ”<<value.second<<endl;
```

Auto Keyword

```
Vector<int>::iterator it;
For(it=v.begin();it!=v.end();it++)
Cout<<(*it)<<endl;

Vector<pair<int,int>>::iterator it;
For(it=v.begin();it!=v.end();it++)
Cout<<(*it).first<<“ ”<<(*it).second<<endl;

For(auto it=v.begin();it!=v.end();it++)
Cout<<(*it)<<endl;
```

Maps in C++ STL

All the keys in a map are unique. Duplicate keys can not be inserted. Instead, trying so will replace the existing key value pair.

```
Int main(){
Map<int,string>m1;
M1[1]=“abc”;
M1[2]=“cde”;
M1[3]=“def”;
M1[4]=“feg”;
M1[3]=“rrr”;
}
```

Ordered Maps

- Unique keys are stored.
- Keys are stored in sorted order.
- Complexity of the insertion operation in O(log(n)).
- Complexity of accessing the elements is also O((log(n))).
- Inserting n elements will take O(nlogn).
- Accessing n elements will take O(nlogn).

Find and Erase

```
Main(){
Map<int,string>m1;
M1[3]=“mayank”;
M1[4]=“mahesh”;
M1[5]=“neeraj”;
Auto it = m1.find(3);
If(it==m1.end)
Cout<<“No value found”;
Else
Cout<<(*it).first<<“”<<(*it).second<<endl;
}
```

Map.erase()

map::erase() is a built-in function in C++ STL which is used to erase element from the container. It can be used to erase keys, elements at any specified position or a given range.

Erase can take two types of arguments either a key or an iterator.

```
Main(){
Map<int,string>m1;
M1[3]=“mayank”;
M1[4]=“mahesh”;
M1[5]=“neeraj”;
Auto it = m1.find(3);
M1.erase(4);
If(it!=m1.end())
m1.erase(it);
```

Set in STL

```
Int main(){
Set<string> S;
s.insert(“abc”);
s.Insert(“pqr”);
s.Insert(“def”);
s.erase(“def”);
}
```