All about STL Maps

```
#include<iostream>
#include<map>
#include<iterator>
using namespace std;


void printMap(map<int,string> &m)
{
    cout<<"Size of map: "<<m.size()<<endl;
    for(auto it:m)
    {

        cout<<it.first<<" "<<it.second<<endl;
    }
}
int main()
{
    map<int,string> m;
    m[1]="utkarsha";
    m.insert({4,"Anushka"});
    m.insert({3,"Chiplunkar"});
    m[2]="chinde";
    m[6]="anu loves uttu";
    m[5]="uttu loves anu";
    m[7];//prints empty key-value pair
    m[2]="stays in miraj";//this will overrite the
value of key 2 as key 2 is already existing

    // cout<<m[1]<<endl;
    // cout<<m[3]<<endl;


// cout<<"\n\n\n";
```

```
//  map<int,string> :: iterator itr;
//  for(itr=m.begin();itr!=m.end();itr++)
//      cout<<itr->first<<" "<<itr->second<<endl;

// cout<<"\n\n\n";

//  for(auto it : m)
//      cout<<it.first<<" "<<it.second<<endl;

    printMap(m);
    cout<<"\n\n\n";


    // auto it=m.find(3);
    auto it =m.find(10);
    // if(it == m.end())
    //  cout<<"no such element present"<<endl;
    // else
    // {
    //  cout<<(*it).first<<" "<<(*it).second<<endl;
    // }


//clear function
    m.clear();
    cout<<"map is cleared"<<endl;
    printMap(m);

//to erase a particular element in map
    // m.erase(4);
    m.erase(it);//does not erased as the element with
key 10 is not present also here the program is
executed but it is not printing the map
    cout<<"erased"<<endl;
```

```
    printMap(m);
    return 0;
}


/*
Insertion in map:-
Insertion in map takes O(log n) cuz the value should
be inserted in sorted order. for searching its sorted
position in tree it requries O(log n) time complexity

The insertion time also depends upon the type of key
i.e if the key is string then we have to compare the
new string with all the existing string to add the new
stirng into its sorted position.
so, now the time complexity is O(n) for comparing
strings * O(log n) to get its sorted position
TE=> O(log n) * s.size()
where s.size() gives size of new string and it is O(n)
OR
TE=> O(log n) * O(n)
where n is the size of new string to be added in map

Accessing key-value in map:-
time complexity is also O(log n) => cuz have to search
in tree


*/
```