Assignment-46:

1. Create a map, insert at least 5 pairs of keys and values and print it.

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
    map<int, int> mp;

    mp.insert({1,11});
    mp.insert({2,12});
    mp.insert({3,13});
    mp.insert({4,14});
    mp.insert({5,15});

    cout<<"key element\n";
    for(auto itr = mp.begin();itr != mp.end(); ++itr)
    {
        cout<<itr->first <<'\t'<<itr->second<<"\n";
    }
    return 0;
}
```

2. Create a map, where insert keys and values as string type and integer type respectively and print it on the screen.

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
```

```
    map<string, int> mp;

    mp.insert({"Rubina",11});
    mp.insert({"Khushi",12});
    mp.insert({"Divya",13});
    mp.insert({"Priya",14});
    mp.insert({"Priyam",15});

    cout<<"key element\n";
    for(auto itr = mp.begin();itr != mp.end(); ++itr)
    {
        cout<<itr->first <<'\t'<<itr->second<<"\n";
    }
    return 0;
}
```

3. Create a map, insert some pairs and print all elements in reverse order using rbegin and rend function.

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
    map<char, int> mp;

    mp.insert({'A',65});
     mp.insert({'B',66});
      mp.insert({'C',67});
       mp.insert({'D',68});
        mp.insert({'E',69});
         mp.insert({'F',70});
```

```
        mp.insert({'G',71});

        cout<<"Element : "<<endl;
        for(auto itr = mp.rbegin(); itr != mp.rend(); ++itr)
        {
            cout<<itr->first<<" "<<itr->second<<endl;
        }
        return 0;
}
```

4. Create a map, and insert some pairs and find one pair out of the inserted pair and replace it with another pair and print map.

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
    map<int, int> mp;
    mp.insert({1,23});
    mp.insert({2,43});
    mp.insert({3,56});
    mp.insert({4,13});
    mp.insert({5,32});

    cout<<"Elements are: "<<endl;

    for(auto itr = mp.begin(); itr != mp.end(); ++itr)
    {
        cout<<itr->first<<" "<<itr->second<<endl;
    }
```

```
    mp[3] = 100;
    mp[5] = 234;

    cout<<"After Replace: "<<endl;

    for(auto itr = mp.begin(); itr != mp.end(); ++itr)
    {
        cout<<itr->first<<" "<<itr->second<<endl;
    }
    return 0;
}
```

5. Create a map, insert some pairs and Find the occurrence of each pair and print it on the screen

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
    map<int, int> mp;

    mp.insert({1,11});
    mp.insert({2,12});
    mp.insert({3,13});
    mp.insert({4,14});
    mp.insert({5,15});

    cout<<"key element\n";
    for(auto itr = mp.begin();itr != mp.end(); ++itr)
    {
```

```
        cout<<itr->first <<'\t'<<itr->second<<"\n";
    }
    return 0;
}
```

6. Create a map, use a member function to tell whether a map is empty or not and then insert some pairs into the map and find the size of map.

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

int main()
{
        map<char, int> mymap;
        mymap['a'] = 1;
        mymap['b'] = 2;
        if (mymap.empty())
        {
           cout << "True";
        }
        else
         {
                cout << "False";
         }
        return 0;
}
```

7. Sort a given map in descending order based on values instead of keys in C++ STL. Key value 1 6 2 8 6 3 8 2

```
#include <bits/stdc++.h>
using namespace std;

int main()
{

	map<int, string> mymap;


	mymap.insert({10, "queen"});
	mymap.insert({20, "rose"});
	mymap.insert({5, " lion"});

	map<int, string>::iterator it;
	for (it = mymap.begin(); it != mymap.end(); it++)
		cout << "(" << (*it).first << ", " << (*it).second
			<< ")" << endl;

	return 0;
}
```

8. Given a string s of length N, containing digits written in words but in jumbled form, the task is to find out the digits present in the string in word form and arrange them in sorted order using map Example: Input: s = “ozerotwneozero” Output: 0012 Explanation: The string can be arranged as “zerozeroonetwo”. Therefore the digits are 0, 0, 1 and 2.

```
#include <bits/stdc++.h>
using namespace std;

string findNumber(string S, int N)
```

```
{
	string ans = "";

	map<char, int> mp;
	for (int i = 0; i < N; i++) {
		mp[S[i]]++;
	}

	while (mp['z'] && mp['e'] && mp['r']
		&& mp['o']) {
		mp['z']--;
		mp['e']--;
		mp['r']--;
		mp['o']--;
		ans += '0';
	}
	while (mp['o'] && mp['n'] && mp['e']) {
		mp['o']--;
		mp['n']--;
		mp['e']--;
		ans += '1';
	}
	while (mp['t'] && mp['w'] && mp['o']) {
		mp['t']--;
		mp['w']--;
		mp['o']--;
		ans += '2';
	}
	while (mp['t'] && mp['h'] && mp['r']
		&& mp['e'] && mp['e']) {
		mp['t']--;
```

```
        mp['h']--;
        mp['r']--;
        mp['e']--;
        mp['e']--;
        ans += '3';
    }
    while (mp['f'] && mp['o'] && mp['u']
        && mp['r']) {
        mp['f']--;
        mp['o']--;
        mp['u']--;
        mp['r']--;
        ans += '4';
    }
    while (mp['f'] && mp['i'] && mp['v']
        && mp['e']) {
        mp['f']--;
        mp['i']--;
        mp['v']--;
        mp['e']--;
        ans += '5';
    }
    while (mp['s'] && mp['i'] && mp['x']) {
        mp['s']--;
        mp['i']--;
        mp['x']--;
        ans += '6';
    }
    while (mp['s'] && mp['e'] && mp['v']
        && mp['e'] && mp['n']) {
        mp['s']--;
        mp['e']--;
```

```
            mp['v']--;
            mp['e']--;
            mp['n']--;
            ans += '7';
        }
        while (mp['e'] && mp['i'] && mp['g']
            && mp['h'] && mp['t']) {
            mp['e']--;
            mp['i']--;
            mp['g']--;
            mp['h']--;
            mp['t']--;
            ans += '8';
        }
        while (mp['n'] && mp['i'] && mp['n']
            && mp['e']) {
            mp['n']--;
            mp['i']--;
            mp['n']--;
            mp['e']--;
            ans += '9';
        }
        return ans;
}

int main()
{
        string s = "zerootwneozero";
        int N = s.size();

        cout << findNumber(s, N);
        return 0;
```

}

9. Given two maps map1 and map2 having a string as the key and arrays of integers as values, the task is to merge them in one map such that if a key is common in both the maps, the respective arrays should be merged. Example: Input: map1 = { (“key1”, {0, 1}), (“key2”, {0, 1}) }, map2 = { (“key2”, {1, 2}) }; Output: { (key1, {0, 1}), (key2, {0, 1, 2}) } Explanation: After merging key1 array will become {0, 1} and for key2 after merging array will become {0, 1, 2}

```
#include <bits/stdc++.h>
using namespace std;

vector<int> mergeArrays(vector<int>& a, vector<int>& b,
                                        int n, int m)
{
        vector<int> mergedArray;

        map<int, bool> mp;

        for (int i = 0; i < n; i++)
                mp[a[i]] = true;

        for (int i = 0; i < m; i++)
                mp[b[i]] = true;

        for (auto i : mp)
                mergedArray.push_back(i.first);
        return mergedArray;
}
```

```
map<string, vector<int> >
mergeMap(map<string, vector<int> >& map1,
         map<string, vector<int> >& map2)
{
    map<string, vector<int> > map3;
    map3.insert(map1.begin(), map1.end());

    for (auto itr : map2) {
        if (map3.find(itr.first) == map3.end())
            map3.insert({ itr.first, itr.second });
        else {
            auto temp_itr = map3.find(itr.first);
            vector<int> arr = mergeArrays(
                itr.second, temp_itr->second,
                itr.second.size(),
                temp_itr->second.size());
            map3[itr.first] = arr;
        }
    }
    return map3;
}


int main()
{
    map<string, vector<int> > map1, map2, map3;
    map1.insert({ "key1", { 0, 1 } });
    map1.insert({ "key2", { 0, 1 } });
    map2.insert({ "key2", { 1, 2 } });
```

```
	map3 = mergeMap(map1, map2);

	for (auto itr : map3) {
		cout << "\"" << itr.first << "\", { ";
		for (auto x : itr.second)
			cout << x << " ";
		cout << "}\n";
	}
	return 0;
}
```

10. Given a positive integer N, the task is to check whether N can be represented as the difference between two positive perfect cubes or not. If found to be true, then print “Yes”. Otherwise, print “No” using a map. Example: Input: N = 124 Output: Yes Explanation: Since 124 can be represented as (125 – 1) = (53 – 13). Therefore, print Yes

```
#include <bits/stdc++.h>
using namespace std;

void differenceOfTwoPerfectCubes(int N)
{
	map<int, int> cubes;

	for (int i = 1;
		(i * i * i) - ((i - 1) * (i - 1) * (i - 1)) <= N;
		i++) {

		cubes[i * i * i] = 1;
	}
```

```
	map<int, int>::iterator itr;


	for (itr = cubes.begin(); itr != cubes.end(); itr++) {


		int firstNumber = itr->first;
		int secondNumber = N + itr->first;
		if (cubes.find(secondNumber) != cubes.end()) {
			cout << "Yes";
			return;
		}
	}
	cout << "No";
}

int main()
{
	int N = 124;
	differenceOfTwoPerfectCubes(N);

	return 0;
}
```