Q1. Create a map, insert at least 5 pairs of keys and values and print it.

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

int main() {

    map <string, int> m;

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 20));
    m.insert(make_pair("Naitik", 30));
    for(pair<string, int> x : m)cout<<x.first<<" : "<<x.second<<endl;
}
```

Q2. Create a map, where insert keys and values as string type and integer type respectively and print it on the screen.

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

int main() {

    map <string, int> m;

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 20));
    m.insert(make_pair("Naitik", 30));
    for(pair<string, int> x : m)cout<<x.first<<" : "<<x.second<<endl;
}
```

Q3. Create a map, insert some pairs and print all elements in reverse order using rbegin and rend function.

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

int main() {

    map <string, int> m;
    map <string, int>::reverse_iterator it = m.rbegin();

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 40));
    m.insert(make_pair("Naitik", 50));

    while(it != m.rend())
    {
        cout<<(*it).first<<" : "<<(*it).second<<endl;
        it++;
    }
}
```

Q4. 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 <iostream>
#include <map>
#include <iterator>
using namespace std;

int main() {

    map <string, int> m;
    map <string, int>::iterator it;

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 40));
    m.insert(make_pair("Naitik", 50));

    string key;

    cout<<"Enter key = ";
    getline(cin, key);

    it = m.find(key);

    if(it != m.end())
    {
        string tmp = (*it).first;
        int t = (*it).second;
        int v;

        m.erase(it->first);

        cout<<"Enter new key = ";
        getline(cin, key);

        cout<<"Enter new value = ";
        cin>>v;

        m.insert(make_pair(key, v));
    }

    for(pair<string, int> x : m)cout<<x.first<<" : "<<x.second<<endl;
}
```

Q5. Create a map, insert some pairs and Find the occurrence of each pair and print it on the screen.)

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

int main() {

    map <string, int> m;

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 20));
    m.insert(make_pair("Naitik", 30));

    for(const pair<string, int> &x : m)cout<<"Occurrence of "<<x.first<<" : "<<x.second<<" is 1"<<endl;
}
```

Q6. 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 <string, int> m;

    if(m.empty())
        cout<<"map is empty"<<endl;
    else
        cout<<"map is not empty"<<endl;

    m.insert(make_pair("Yash", 10));
    m.insert(make_pair("Ravi", 20));
    m.insert(make_pair("Rahul", 30));
    m.insert(make_pair("Vrushabh", 20));
    m.insert(make_pair("Naitik", 30));

    cout<<"pair inserted and size of map is "<<m.size();
}
```

Q7. 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 <iostream>
#include <map>
#include <vector>
#include <algorithm>
using namespace std;

int main()
{
    map <int, int> m {{1, 6}, {2, 8}, {6, 3}, {8, 2}};

    vector <pair<int, int>> v(m.begin(), m.end());

    sort(v.begin(), v.end(), [](pair<int, int> &a, pair<int, int> &b){return a.second > b.second;});

    for(pair<int, int> &x : v) cout<<x.first<<" : "<<x.second<<endl;
}
```

Q9. 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 <iostream>
#include <map>
#include <string>
#include <array>
#include <set>
using namespace std;

int main()
{
   map <string, array <int, 2>> map1 = {{"key1",{0, 1}}, {"key2",{0, 1}}};
   map <string, array <int, 2>> map2 = {{"key2", {1, 2}}};

   map <string, set<int>> marge;

   map <string, array<int, 2>>::iterator it1 = map1.begin();

   //checking map1 key is exist or not in map2
   for(; it1 != map1.end(); it1++)
   {
      int flag = 1;
      map <string, array<int, 2>>::iterator it2 = map2.begin();

      for(; it2 != map2.end(); it2++)
      {
         if(it1->first == it2->first) //if key is same then copy data into marge map
         {
            set <int> s;

            for(int i = 0; i < 2; i++)
            {
               s.insert(it1->second[i]);
               s.insert(it2->second[i]);
            }

            string key = it1->first;
            marge.insert(make_pair(key, s));

            flag = 1;
            break;
         }
      }
      if(flag) //if key is not same then copy only m1 data
      {
          set <int> s;
         for(int i = 0; i < 2; i++)
         {
            s.insert(it1->second[i]);
            }
            string key = it1->first;
            marge.insert(make_pair(key, s));
        }
    }

    for(auto it1 = marge.begin(); it1 != marge.end(); it1++)
    {
        cout<<it1->first<<" : ";
        for(int x : it1->second) cout<<x<<" ";
        cout<<endl;
    }
}
```

Q10. 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 <iostream>
#include <map>
#include <cmath>
using namespace std;

bool diff(int d, map <int, int> &cube)
{
   for(const pair<int, int> &y : cube)
   {
      if(d == y.second)//d = 2
      {
         return true;
      }
      if(d < y.second)
      {
         return false;
      }
   }
   return false;
}

int main()
{
   int N, a, b;
   a = b = N = 0;

   cout<<"Enter N = ";
   cin>>N;

   map <int, int> cube;

   //insert cube in map
   for(int i = 0; i < 30; i++)
   {
      cube[i] = pow(i, 3);
   }

   //find a
   for(const pair<int, int> &x : cube)
   {
      if(x.second == N)
      {
         cout<<"Yes";
         break;
      }
      if( (x.second > N))//125>61
      {
            if(diff(x.second - N, cube))
            {
                cout<<"Yes";
                break;
            }
            else
            {
                if(((x.second - N) > N))
                {
                    cout<<"No";
                    break;
                }
            }

        }
    }
}
```