# Map in C++ Standard Template Library (STL)

Maps are associative containers that store elements in a mapped fashion. Each element has a key value and a mapped value. No two mapped values can have same key values.

Functions associated with Map:

begin() – Returns an iterator to the first element in the map

end() – Returns an iterator to the theoretical element that follows last element in the map

size() – Returns the number of elements in the map

max_size() – Returns the maximum number of elements that the map can hold

empty() – Returns whether the map is empty

pair insert(keyvalue,mapvalue) – Adds a new element to the map

erase(iterator position) – Removes the element at the position pointed by the iterator

erase(const g)- Removes the key value ‘g’ from the map

clear() – Removes all the elements from the map

key_comp() / value_comp() – Returns the object that determines how the elements in the map are ordered (‘<' by default)

find(const g) – Returns an iterator to the element with key value ‘g’ in the map if found, else returns the iterator to end

count(const g) – Returns the number of matches to element with key value ‘g’ in the map

lower_bound(const g) – Returns an iterator to the first element that is equivalent to mapped value with key value ‘g’ or definitely will not go before the element with key value ‘g’ in the map

upper_bound(const g) – Returns an iterator to the first element that is equivalent to mapped value with key value ‘g’ or definitely will go after the element with key value ‘g’ in the map

```
#include <iostream>
#include <map>
#include <iterator>

using namespace std;

int main()
{
    map <int, int> gquiz1;        // empty map container

    // insert elements in random order
    gquiz1.insert(pair <int, int> (1, 40));
    gquiz1.insert(pair <int, int> (2, 30));
    gquiz1.insert(pair <int, int> (3, 60));
    gquiz1.insert(pair <int, int> (4, 20));
    gquiz1.insert(pair <int, int> (5, 50));
    gquiz1.insert(pair <int, int> (6, 50));
    gquiz1.insert(pair <int, int> (7, 10));

    // printing map gquiz1
    map <int, int> :: iterator itr;
    cout << "\nThe map gquiz1 is : \n";
    cout << "\tKEY\tELEMENT\n";
```



```
    }
```

```
    cout << endl;

    // assigning the elements from gquiz1 to gquiz2
    map <int, int> gquiz2(gquiz1.begin(), gquiz1.end());

    // print all elements of the map gquiz2
    cout << "\nThe map gquiz2 after assign from gquiz1 is : \n";
    cout << "\tKEY\tELEMENT\n";
    for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr)
    {
        cout  <<  '\t' << itr->first
              <<  '\t' << itr->second << '\n';
    }
    cout << endl;

    // remove all elements up to element with key=3 in gquiz2
    cout << "\ngquiz2 after removal of elements less than key=3 : \n";
    cout << "\tKEY\tELEMENT\n";
    gquiz2.erase(gquiz2.begin(), gquiz2.find(3));
    for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr)
    {
        cout  <<  '\t' << itr->first
              <<  '\t' << itr->second << '\n';
    }

    // remove all elements with key = 4
    int num;
    num = gquiz2.erase (4);
    cout << "\ngquiz2.erase(4) : ";
    cout << num << " removed \n" ;
    cout << "\tKEY\tELEMENT\n";
    for (itr = gquiz2.begin(); itr != gquiz2.end(); ++itr)
    {
        cout  <<  '\t' << itr->first
              <<  '\t' << itr->second << '\n';
    }

    cout << endl;

    //lower bound and upper bound for map gquiz1 key = 5
    cout << "gquiz1.lower_bound(5) : " << "\tKEY = ";
    cout << gquiz1.lower_bound(5)->first << '\t';
    cout << "\tELEMENT = " << gquiz1.lower_bound(5)->second << endl;
    cout << "gquiz1.upper_bound(5) : " << "\tKEY = ";
    cout << gquiz1.upper_bound(5)->first << '\t';
    cout << "\tELEMENT = " << gquiz1.upper_bound(5)->second << endl;

    return 0;

}
```

Run on IDE

The output of the above program is :

```
The map gquiz1 is :
    KEY    ELEMENT
    1    40
    2    30
    3    60
    4    20
    5    50
    6    50
    7    10


The map gquiz2 after assign from gquiz1 is :
    KEY    ELEMENT
    1    40
    2    30
    3    60
    4    20
    5    50
    6    50
    7    10
```

```
    4    20
    5    50
    6    50
    7    10

gquiz2.erase(4) : 1 removed
    KEY    ELEMENT
    3    60
    5    50
    6    50
    7    10

gquiz1.lower_bound(5) :     KEY = 5        ELEMENT = 50
gquiz1.upper_bound(5) :     KEY = 6        ELEMENT = 50
```

**Useful Links :**

- Recent Articles on C++ Map
- map vs unordered_map in C++
- Inserting elements in std::map (insert, emplace and operator [])
- Searching in a map using std::map functions in C++
- C++ map having key as a user define data type
- map::at() and map::swap() in C++ STL
- map::clear() in C++ STL
- map::at() in C++ STL
- map::operator[] in C++ STL
- map::begin() and end() in C++ STL
- map::empty() in C++ STL
- map::size() in C++ STL

C++ Programming Language Tutorial | Map in C++ STL | GeeksforGeeks

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

**Article Tags :** C++ cpp-containers-library cpp-map STL

Login to Improve thisArticle

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

## Recommended Posts:

Multimap in C++ Standard Template Library (STL)
unordered_map in STL and its applications
The C++ Standard Template Library (STL)
Set in C++ Standard Template Library (STL)
Pair in C++ Standard Template Library (STL)
list max_size() function in C++ STL
asinh() function in C++ STL
atanh() function in C++ STL
multimap equal_range() in C++ STL
Check if X can give change to every person in the Queue

(Login to Rate)

2.8 Average Difficulty : **2.8/5.0**
Based on **34** vote(s)

Basic Easy Medium Hard Expert

Add to TODO List
Mark as DONE

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments Share this post!