Map in C++ Standard Template Library (STL)

Difficulty Level : Medium • Last Updated : 01 Feb, 2023

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 the same key values.

Some basic 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 the 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.

Begin() function :

C++

```cpp
#include <iostream>
#include <map>

int main()
{
  // Create a map of strings to integers
  std::map<std::string, int> map;

  // Insert some values into the map
  map["three"] = 3;
```

Read Discuss(20) Courses Practice Video

```
  // Iterate through the map and print the elements
  while (it != map.end())
  {
    std::cout << "Key: " << it->first << ", Value: " << it->second << std::endl;
    ++it;
  }

  return 0;
}
```

Output

```
Key: one, Value: 1
Key: three, Value: 3
Key: two, Value: 2
```

Time complexity: O(n) //where n is the size of map.

Auxiliary Space: O(n)

end ()function:

See a successful future with C++
A COMPLETE BEGINNER TO ADVANCED C++ COURSE
25+ Hours Of Content
150+ Curated Problems
24*7 Doubt Support
Explore Course

Data Structures and Algorithms Interview Preparation Data Science Topic-wise Practice C

C++

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

art Your Coding Journey Now!

```
  // Create a map of strings to integers
  std::map<std::string, int> map;
```

```
  map["two"] = 2;
  map["three"] = 3;

  // Get an iterator pointing to the first element in the map
  std::map<std::string, int>::iterator it = map.begin();

  // Iterate through the map and print the elements
  while (it != map.end())
  {
    std::cout << "Key: " << it->first << ", Value: " << it->second << std::endl;
    ++it;
  }

  return 0;
}
```

Output

```
Key: one, Value: 1
Key: three, Value: 3
Key: two, Value: 2
```

Time complexity: O(n) //where n is the size of map.

Auxiliary Space: O(n)

Size function:

C++

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

int main()
{
  // Create a map of strings to integers
  std::map<std::string, int> map;

  // Insert some values into the map
  map["one"] = 1;
  map["two"] = 2;
```

```
    // Print the size of the map
    std::cout << "Size of map: " << map.size() << std::endl;
```

```
}
```

Output

```
Size of map: 3
```

Time complexity: O(1).

Implementation:

CPP

```
// CPP Program to demonstrate the implementation in Map
// divyansh mishra --> divyanshmishra101010
#include <iostream>
#include <iterator>
#include <map>
using namespace std;

int main()
{

    // empty map container
    map<int, int> gquiz1;

    // 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[7]=10;     // another way of inserting a value in a map


    // printing map gquiz1
    map<int, int>::iterator itr;
    cout << "\nThe map gquiz1 is : \n";
    cout << "\tKEY\tELEMENT\n";
    for (itr = gquiz1.begin(); itr != gquiz1.end(); ++itr) {
        cout << '\t' << itr->first << '\t' << itr->second
             << '\n';
    // assigning the elements from gquiz1 to gquiz2
    map<int, int> gquiz2(gquiz1.begin(), gquiz1.end());
```

```
         << " 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;
}
```

Output

Read Discuss(20) Courses Practice Video

```
    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


gquiz2 after removal of elements less than key=3 :
    KEY    ELEMENT
    3    60
    4    20
    5    50
    6    50
    7    10

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

art Your Coding Journey Now!

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

Auxiliary space: O(n)

Example:

C++

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

int main()
{
  // Create a map of strings to integers
  std::map<std::string, int> map;

  // Insert some values into the map
  map["one"] = 1;
  map["two"] = 2;
  map["three"] = 3;

  // Print the values in the map
  std::cout << "Key: one, Value: " << map["one"] << std::endl;
  std::cout << "Key: two, Value: " << map["two"] << std::endl;
  std::cout << "Key: three, Value: " << map["three"] << std::endl;

  // Check if a key is in the map
  if (map.count("four") > 0)
  {
    std::cout << "Key 'four' is in the map" << std::endl;
  }
  else
  {
    std::cout << "Key 'four' is not in the map" << std::endl;
  }

  return 0;
}
```

Output

```
Key: one, Value: 1
```

```
Key 'four' is not in the map
```

Function	Definition
map::insert()	Insert elements with a particular key in the map container -> O(log n)
map:: count()	Returns the number of matches to element with key-value 'g' in the map. -> O(log n)
map equal_range()	Returns an iterator of pairs. The pair refers to the bounds of a range that includes all the elements in the container which have a key equivalent to k.
map erase()	Used to erase elements from the container -> O(log n)
map rend()	Returns a reverse iterator pointing to the theoretical element right before the first key-value pair in the map(which is considered its reverse end).
map rbegin()	Returns a reverse iterator which points to the last element of the map.
map find()	Returns an iterator to the element with key-value 'g' in the map if found, else returns the iterator to end.
map crbegin() and crend()	crbegin() returns a constant reverse iterator referring to the last element in the map container. crend() returns a constant reverse iterator pointing to the theoretical element before the first element in the map.
map cbegin() and	cbegin() returns a constant iterator referring to the first

Function	Definition
	in the multimap.
map emplace()	Inserts the key and its element in the map container.
map max_size()	Returns the maximum number of elements a map container can hold -> O(1)
map upper_bound()	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
map operator=	Assigns contents of a container to a different container, replacing its current content.
map lower_bound()	Returns an iterator to the first element that is equivalent to the mapped value with key-value 'g' or definitely will not go before the element with key-value 'g' in the map -> O(log n)
map emplace_hint()	Inserts the key and its element in the map container with a given hint.
map value_comp()	Returns the object that determines how the elements in the map are ordered ('<' by default).
map key_comp()	Returns the object that determines how the elements in the map are ordered ('<' by default).
map::size()	Returns the number of elements in the map.
map::empty()	Returns whether the map is empty
map::begin() and end()	begin() returns an iterator to the first element in the map. end() returns an iterator to the theoretical element that follows the

Function	Definition
	position given inside the operator.
map::clear()	Removes all the elements from the map.
map::at() and map::swap()	at() function is used to return the reference to the element associated with the key k. swap() function is used to exchange the contents of two maps but the maps must be of the same type, although sizes may differ.

Recent Articles on Map

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

Like 402

Previous Next

Related Articles

1. List in C++ Standard Template Library (STL)
2. Multimap in C++ Standard Template Library (STL)
3. Multiset in C++ Standard Template Library (STL)

art Your Coding Journey Now!

5. The C++ Standard Template Library (STL)

Read Discuss(20) Courses Practice Video

7. Queue in C++ Standard Template Library (STL)

8. Containers in C++ STL (Standard Template Library)

9. Pair in C++ Standard Template Library (STL)

10. Sort in C++ Standard Template Library (STL)

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : anshikajain26, divyanshmishra101010, rathoadavinash

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

Practice Tags : CPP, STL

Improve Article Report Issue

art Your Coding Journey Now!

Sector-136, Noida, Uttar Pradesh - 201305

Read Discuss(20) Courses Practice Video

Company

About Us
Careers
In Media
Contact Us
Privacy Policy
Copyright Policy
Advertise with us

Learn

DSA
Algorithms
Data Structures
SDE Cheat Sheet
Machine learning
CS Subjects
Video Tutorials
Courses

News

Top News
Technology
Work & Career
Business
Finance
Lifestyle
Knowledge

Languages

Python
Java
CPP
Golang
C#
SQL
Kotlin

Web Development

Web Tutorials
Django Tutorial
HTML
JavaScript
Bootstrap
ReactJS
NodeJS

Contribute

Write an Article
Improve an Article
Pick Topics to Write
Write Interview Experience
Internships
Video Internship

art Your Coding Journey Now!