How to Iterate over a map in C++

Varun September 17, 2016 std::map, STL 1 Comment

In this article we will discuss 3 different ways to Iterate over a map in C++.

Suppose we have a map of string and int as key-value pair i.e.

```
std::map<std::string, int> mapOfWordCount;
```

Now let's see how to iterate over this map in 3 different ways i.e.

Iterate over a map using STL Iterator

First of all, create an iterator of std::map and initialize it to the beginning of map i.e.

```
std::map<std::string, int>::iterator it = mapOfWordCount.begin()
```

Now, let's iterate over the map by incrementing the iterator until it reaches the end of map. Also, map internally stores element in a std::pair format, therefore each iterator object points to an address of pair.

Access key from iterator using,

```
it->first
```

Vector List Deque
Set Map MultiMap
STL Algorithms

1.) What's std::vector and why to use it?
2.) Different ways to initialize a vector
3.) How does std::vector works internally
4.) User Defined Objects & std::vector
5.) How to use vector efficiently in C++?
6.) std::vector and Iterator Invalidation
7.) Remove repeated elements from a vector
8.) Fill a vector with random numbers
9.) Hidden cost of std::vector
10.) Adding elements in Vector

Access value from iterator using,

```
it->second
```

Complete example of iterating over a map using stl iterator is as follows,

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

int main() {

    std::map<std::string, int> mapOfWordCount;
    // Insert Element in map
    mapOfWordCount.insert(std::pair<std::string, int>("first", 1
    mapOfWordCount.insert(std::pair<std::string, int>("second",
    mapOfWordCount.insert(std::pair<std::string, int>("third", 3
    mapOfWordCount.insert(std::pair<std::string, int>("third", 4
    mapOfWordCount.insert(std::pair<std::string, int>("third", 5

    // Create a map iterator and point to beginning of map
    std::map<std::string, int>::iterator it = mapOfWordCount.beg

    // Iterate over the map using Iterator till end.
    while (it != mapOfWordCount.end())
    {
        // Accessing KEY from element pointed by it.
        std::string word = it->first;

        // Accessing VALUE from element pointed by it.
        int count = it->second;

        std::cout << word << " :: " << count << std::endl;

        // Increment the Iterator to point to next entry
        it++;
    }
    return 0;
}
```

Output:

```
first :: 1
second :: 2
third :: 3
```

Iterating over the map using C++11 range based for loop

C++11 provides a range based for loop, we can also use that to iterate over the map. In that case we don't need iterate and it will take less coding. Check out the following example,

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

int main() {

    std::map<std::string, int> mapOfWordCount;
    // Insert Element in map
    mapOfWordCount.insert(std::pair<std::string, int>("first", 1
    mapOfWordCount.insert(std::pair<std::string, int>("second",
    mapOfWordCount.insert(std::pair<std::string, int>("third", 3
    mapOfWordCount.insert(std::pair<std::string, int>("third", 4
    mapOfWordCount.insert(std::pair<std::string, int>("third", 5

    // Create a map iterator and point to beginning of map
    std::map<std::string, int>::iterator it = mapOfWordCount.beg

    // Iterate over the map using c++11 range based for loop
    for (std::pair<std::string, int> element : mapOfWordCount) {
        // Accessing KEY from element
        std::string word = element.first;
        // Accessing VALUE from element.
        int count = element.second;
        std::cout << word << " :: " << count << std::endl;
    }

    return 0;
}
```

Output:

```
first :: 1
second :: 2
third :: 3
```

Above example is using c++11 feature. So, to compile it on linux use following command,

g++ -std=c++11 example.cpp

Iterating over the map using std::for_each and lambda function

We can also use an stl algorithm std::for_each to iterate over the map. It will iterate on each of the map entry and call the callback provided by us. In below example we will use a lambda function as callback. Lambda function will receive each of the map entry in a pair. Checkout complete example as follows,

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

```
#include <algorithm>

int main() {

    std::map<std::string, int> mapOfWordCount;
    // Insert Element in map
    mapOfWordCount.insert(std::pair<std::string, int>("first", 1
    mapOfWordCount.insert(std::pair<std::string, int>("second",
    mapOfWordCount.insert(std::pair<std::string, int>("third", 3
    mapOfWordCount.insert(std::pair<std::string, int>("third", 4
    mapOfWordCount.insert(std::pair<std::string, int>("third", 5

    // Create a map iterator and point to beginning of map
    std::map<std::string, int>::iterator it = mapOfWordCount.beg

    // Iterate over a map using std::for_each and Lambda functio
        std::for_each(mapOfWordCount.begin(), mapOfWordCount.end
                []([std::pair<std::string, int> element){
                    // Accessing KEY from element
                    std::string word = element.first;
                    // Accessing VALUE from element.
                    int count = element.second;
                    std::cout<<word<<" :: "<<count<<std::endl;
        });

    return 0;
}
```


Output:

```
first :: 1
second :: 2
third :: 3
```

Above example is using c++11 feature. So, to compile it on linux use following command,

g++ -std=c++11 example.cpp

Click Here to Subscribe for more Articles / Tutorials like this.

Related Posts:

- C++ Map Insert Example
- How to iterate a map in reverse order – C++
- How to Sort a Map by Value in C++
- How to iterate over an unordered_map in C++11
- Different ways to insert elements in an unordered_map
- How to Insert elements in an unordered_set in C++11

- multimap Example and Tutorial in C++
- C++11 'auto' Tutorial and Examples
- How to Access Element by index in a Set | C++
- C++ : Different ways to insert elements in Set
- How to copy all Values from a Map to a Vector in C++
- Different ways to iterate over a set in C++
- C++ : Different Ways to iterate over a List of objects
- Unordered_map Usage Tutorial and Example
- How to Erase / Remove an element from an unordered_map
- Different Ways to initialize an unordered_map
- C++ : How to get element by index in List
- C++ : How to find an element in vector and get its index ?
- Finding all values for a key in multimap using...
- C++ : How to find duplicates in a vector ?
- C++ map : Erase element by key or Iterator or Range
- How check if a given key exists in a Map | C++
- How to remove elements from a List while Iterating
- How to iterate over a HashSet in Java
- How to find an element in unordered_map

1 Comment Already

Leave a Reply

Your email address will not be published. Required fields are marked *

This site

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment

uses Akismet to reduce spam. Learn how your comment data is processed.

« How to pass an ArrayList to varargs method

C++ Map Insert Example »

Python : List

Check if an item exists in List

Check if a list contains all the elements of other list

Create a List and initialize with values

How to Iterate over a List

Insert an element at specific index in List

Sort a list of tuples by 2nd Item

Sort a list of strings

Add an element in list | append() vs extend()

Check if all elements in a List are same

C++11 – Utilities

std::bind

auto specifier

Variadic Templates

C++11 – Unordered Set

1.) unordered_set Basic Example

2.) Initializing an unordered_set

3.) Inserting elements in an unordered_set

4.) Searching an element in unordered_set

5.) unordered_set – Custom Hasher &

Design Patterns

Behavioral Design Patterns

- Observer Design Pattern
- State Design Pattern
- Strategy Design Pattern

Structural Design Patterns

STL – Vector

1.) What's std::vector and why to use it?

2.) Different ways to initialize a vector

3.) How does std::vector works internally

4.) User Defined Objects & std::vector

5.) How to use vector efficiently in C++?

6.) std::vector and Iterator Invalidation

7.) Remove repeated elements from a vector

8.) Fill a vector with random numbers

9.) Hidden cost of std::vector

Python : Dictionary

Python Strings

Java – Hashmap

C++11 – UnorderedMap

C++11 Smart Pointers

C++11 Multithreading

Pointers

Callbacks in C++

C++ Strings

C++ Memory Manangement

STL – Deque

STL – List

STL – Set

What is Hashing and Hash Table?

Associating Multiple values with same Key

Remove elements while Iterating

Update the value of an existing key

Get all keys by a value in HashMap

Java – HashSet

What is Hashing and Hash Table?

Create and add elements in a HashSet

Iterate over a HashSet

Search for an element in HashSet

Merge two HashSets

Initializing HashSet from an Array

Convert a HashSet into an Array

Merge an Array in a HashSet

Java Interview Questions

Method Overriding Tutorial

Overriding with Different Return Type

Calling Base class's overridden method

Preventing Method Overriding

Need of preventing Method Overriding

Part 6 : Need of Event Handling

Part 7: Condition Variables

Part 8: std::future and std::promise

Part 9: std::async Tutorial & Example

Part 10: std::packaged_task<> Tutorial

C++11 Rvalue References

lvalue vs rvalue

Is rvalue immutable in C++?

What is rvalue reference in C++11

Move Contsructor

new and delete operator

delete vs []delete

Out Of Memory Errors

Overload new & delete

Restrict Dynamic Deletion

Placement new operator

Delete 'this' pointer

Polymorphism

Virtual Functions

vTable and vPointer

8.) Erase elements while Iterating & Generic erase_if()

STL – Map

1.) std::map Usage Detail with examples

2.) std::map and Comparator

3.) std::map & User defined class objects as keys

4.) Set vs Map

5.) How to Iterate over a map in C++

6.) Map Insert Example

7.) Iterate a map in reverse order

8.) Check if a key exists in a Map

9.) Search by value in a Map

10.) Erase by Key | Iterators

11.) C++ Map : Operator []

12.) Erase by Value or callback

13.) copy all Values from a Map to vector

STL Multimap

MultiMap Example and Tutorial

multimap::equals_range – Tutorial

STL Algorithms

std::sort Tutorial & Example

std::unique Tutorial & Example

1.) Using std::find & std::find_if with User Defined Classes

2.) Iterating over a range of User Defined objects and calling

member function using std::for_each

Terms and Conditions

Terms and Conditions Policy