C++11 : std::any_of() Examples and Tutorial

Varun September 21, 2017 C++ 11 No Comment

In this article we will discuss how to use STL Algorithm std::any_of() with both lambda function and function pointer.

std::any_of is an STL algorithm introduced in c++11.

Need of std::any_of()

This STL algorithm is useful when you have a range of element and you want to check if any of the given elements in range satisfies a given condition.

std::any_of() Usage Deatils

std::any_of() accepts a range of elements and a Unary Predicate (callback) as an argument.

```
1 bool any_of (InputIterator start, InputIterator end, UnaryPredic
```

std::any_of() iterates over the given range and for each elements calls the given callback i.e. Unary Predicate. If for any element then given predicate returns true then it stops the further iteration and return true, else it returns false.

std::any_of() Examples

Std::Tuple
Lambda Functions
C++11 Utilities
Initializer_list
Std::Array
Rvalue References
Smart Pointers
Multithreading
Unordered_Set
Unordered_Map

c++11 : std::tuple Tutorial
c++11 : std::make_tuple Tutorial

How to use std::any_of() with two different type of callbacks i.e Lambda Function and Function pointer.

Using std::any_of() with Lambda Function

Suppose We have a vector of strings i.e.

```
std::vector<std::string> vecOfStrs =
    { "Hi", "Hello", "test", "first", "second", "third", "fourth
```

Now we want to check if this vector contains any string with size 4. Let's do this using std::any_of() i.e.

```
/*
Check if vector contains any string with size 4.
*/
bool result = std::any_of(vecOfStrs.begin(), vecOfStrs.end(), []
                                                    return s
                                                    });
```

std::any_of() will iterate through all the strings in vector and for each string in vector it calls the passed lambda function, that checks if size of string is 4. If for any string the lambda function returns true, std::any_of() will stop the further iteration and returns true else it returns false.

Now check if the above vector contains any string that starts with char 'P' i.e.

```
/*
 Check if vector contains any string that starts with char 'P'
 */
result = std::any_of(vecOfStrs.begin(), vecOfStrs.end(), [](cons
                                                    return s
                                                });
```

Using std::any_of() with a Function Pointer

Suppose we have a string i.e.

```
std::string str = "testString";
```

Now let's check if given string contains all lower case letters i.e. not a single upper case char i.e.

```
/*
```

```
 Check if given string contains all lower case letters i.e. not
 */
result = std::any_of(str.begin(), str.end(), ::isupper);
```

Checkout complete example as follows,

```C++
#include <vector>
#include <string>
#include <algorithm>

int main()
{
    std::vector<std::string> vecOfStrs =
    { "Hi", "Hello", "test", "first", "second", "third", "fourth

    /*
     Check if vector contains any string with size 4.
     */
    bool result = std::any_of(vecOfStrs.begin(), vecOfStrs.end()



    std::cout << "vector contains any string with size 4 | Resul

    /*
     Check if vector contains any string that starts with char '
     */
    result = std::any_of(vecOfStrs.begin(), vecOfStrs.end(), [](
                                                        retu
                                                    });

    std::cout<< "vector contains any string that starts with cha

    std::string str = "testString";

    /*
     Check if given string contains all lower case letters i.e.
     */

    result = std::any_of(str.begin(), str.end(), ::isupper);

    std::cout << "str  = " << str << std::endl;
    std::cout<< "Check if given string contains all lower case l
    return 0;
}
```

Output:

```
vector contains any string with size 4 | Result = 1
vector contains any string that starts with char 'P' | Result = (
str  = testString
Check if given string contains all lower case letters | Result =
```

To compile the above code use following command in linux,

g++ –std=c++11 example.cpp

Related Posts:

- C++11 – std::all_of() – Algorithm...
- C++ : Case-insensitive string comparison using STL |...
- C++ : How to read a file line by line into a vector ?
- C++ : Different ways to insert elements in Set
- C++ : How to check if a String Ends With an another...
- C++ : How to compare two vectors | std::equal()...
- C++ : Check if a String starts with an another given String
- C++ : How to find an element in vector and get its index ?
- Converting a String to Upper & Lower Case in...
- Erase elements from a Set while Iterating in C++...
- C++ : strcmp() Tutorial | Comparing strings
- How to copy all Values from a Map to a Vector in C++
- 5 Different ways to Initialize a vector in c++
- Convert First Letter of each word of a String to...
- C++ Map Insert Example
- Find all occurrences of a sub string in a string |...
- How to search by value in a Map | C++
- C++ List – Find | Contains : How to search an...
- How to add an element in Vector using vector::push_back
- C++ : How to get element by index in vector | at()...
- How to Insert elements in an unordered_set in C++11
- How to fill a vector with random numbers in C++
- How to Access Element by index in a Set | C++
- STL Algorithm std::unique Tutorial
- C++ : How to find duplicates in a vector ?

No Comments Yet

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.

« C++ : Check if a String starts with an another given String

How to write data in a CSV file in C++ »

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

C++11 – Utilities

std::bind

auto specifier

Variadic Templates

C++11 – Unordered Set

Advertisement

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

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

Merge / Join two or more lists

Remove Duplicates from a List

Convert a list to string

Remove element from a list by value or Index

Remove multiple elements from list

Python : Dictionary

Creating Dictionaries in Python

Iterating over Dictionaries in Python

Check if a key exists in Dictionary

Get list of all the keys in Dictionary

Get list of all the Values in a Dictionary

Remove multiple keys in Dictionary while Iterating

Remove a key from Dictionary

Add key/value pairs in Dictionary

Find keys by value in Dictionary

Sort a Dictionary by key or Value

Copy a dictionary | Shallow vs Deep Copy

Python Strings

Access characters in string by index in Python

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 & Comparator

6.) Unordered_set & User defined classes

C++11 – UnorderedMap

Basic Usage Detail and Example

Initializing an unordered_map

Searching in unordered_map

Insert elements in unordered_map

Erasing an element

Erase elements while iterating

std::map vs std::unordered_map

C++11 Smart Pointers

shared_ptr<> Tutorial and Examples

shared_ptr and Custom Deletor

shared_ptr vs raw pointer

Create shared_ptr objects carefully

weak_ptr Tutorial | shared_ptr and Cyclic References

unique_ptr<> Tutorial and Examples

C++11 Multithreading

Behavioral Design Patterns

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

Structural Design Patterns

- Composite Design Pattern
- Flyweight Design Pattern

Creational Design Patterns

- Factory Method Design Pattern

Pointers

Pointer vs Refrence

Allocating 2D Array Dynamically

Callbacks in C++

Function Pointers

Function Objects & Functors

C++ Strings

Find and Replace all occurrences of a string

Find all occurrences of a sub string

Case Insensitive string::find

Convert First Letter of each word to Upper Case

Converting a String to Upper & Lower Case

Trim strings in C++

C++ : How to split a string using String and character as Delimiter?

startsWith() Implementation

endsWith() Implementation

Remove Sub Strings from String

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

STL – Deque

1.) What is std::deque and how deque works internally.

2.) deque vs vector : What to choose ?

STL – List

1.) std::list Internals & Usage Details

2.) List vs Vector

3.) Different ways to Initialize a list

4.) Erase elements using iterators

5.) Remove elements while Iterating

6.) Remove elements based on External Criterion

7.) Get element by index in List

8.) Searching an element in std::list

9.) Different Ways to iterate over a List

10.) Sorting a List & custom Comparator

STL – Set

1.) C++ Set basic example and Tutorial

2.) Using std::set with user defined classes

Iterate over the characters in string

How to Replace characters in a string ?

Java – Hashmap

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 1: Three Ways to Create Threads

Part 2: Joining and Detaching Threads

Part 3: Passing Arguments to Threads

Part 4 : Sharing Data & Race Conditions

Part 5 : Fixing Race Conditions using mutex

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

C++ Memory Manangement

Memory Leaks

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

3.) std::set and external Sorting criteria | Comparator

4.) Access Element by index in Set

5.) How to insert elements in Set

6.) How to iterate over a Set

7.) Removing an element from Set

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